Supreme Court, U.S.
FILED
MAY 30 1980
MICHAEL RODAK, JR., CLERK

APPENDIX

In the Supreme Court of the United States

OCTOBER TERM, 1979

No. 79-855

SIDNEY A. DIAMOND, COMMISSIONER OF PATENTS and TRADEMARKS,

Petitioner

—*v.*—

JOHN J. BRADLEY and BENJAMIN S. FRANKLIN

ON WRIT OF CERTIORARI TO THE UNITED STATES COURT OF CUSTOMS AND PATENT APPEALS

PETITION FOR A WRIT OF CERTIORARI FILED DECEMBER 3, 1979
CERTIORARI GRANTED MARCH 17, 1980

In the Supreme Court of the United States

OCTOBER TERM, 1979

No. 79-855

SIDNEY A. DIAMOND, COMMISSIONER OF PATENTS and TRADEMARKS,

Petitioner

—*v.*—

JOHN J. BRADLEY and BENJAMIN S. FRANKLIN

ON WRIT OF CERTIORARI TO THE UNITED STATES COURT OF CUSTOMS AND PATENT APPEALS

INDEX

	Page
Docket Entries	1
Letter of Transcriber dated December 7, 1979	3
Photographs	4
(a) Punch cards of the type utilized by programmers in computer programming	4
(b) Magnetic Tape of the type utilized by programmers in computer programming	5
(c) A computer hardwired board of the type utilized by system designers to design new computer machines	6
(d) Two computer firmware boards of the type utilized by system designers to design new computer machines	
Board (1)	7
Board (2)	8
(e) A MOS memory board of the type utilized for main memories in computer machines	9
Opinion of the United States Court of Customs and Patent Appeals dated July 5, 1979	10
Judgment of the United States Court of Customs and Patent Appeals dated July 27, 1979	20a

	Page
Praecipe for transcript of record dated November 17, 1978	21
Application Serial No. 570,331 (including petition, specification, drawings and declaration as filed, but not including claims)	24
Letter of the Examiner (First Office Action) March 12, 1976	121
Notice of references cited	125
Amendment, August 11, 1976	126
Letter of Examiner (Second Office Action) dated October 27, 1976	150
Letter to Office, December 30, 1976. Amendment under rule 116	153
Letter of Examiner, February 1, 1977. Advisory Action	163
Brief for Appellants, April 19, 1977	165
Examiner's Answer, August 4, 1977	179
Reply brief for Appellants, August 17, 1977	184
Opinion and Decision of Board of Appeals, September 20, 1978	193
Notice and Reasons of Appeal to United States Court of Customs and Patent Appeals, November 17, 1978	204
Rejected Claims	206
Order Granting Certiorari	208

DOCKET ENTRIES

Docket No. 79-533

IN THE MATTER OF THE APPLICATION OF
JOHN J. BRADLEY and BENJAMIN S. FRANKLIN

SWITCH SYSTEM BASE MECHANISM

Board of Appeals
Serial No. 570,331

[Filed December 21, 1978]

DATE	PROCEEDINGS AND ORDERS

Counsel for appellants: Nicholas Prasinos; Ronald T. Reiling Commissioner of Patents & Trademarks

February 12, 1979 PRINTED TRANSCRIPT OF RECORD, filed.

March 8, 1979 BRIEF FOR APPELLANT, filed.

April 9, 1979 Motion of Commissioner of Patents to extend time for filing brief to April 23, 1979, filed.

April 10, 1979 Above motion granted.

April 23, 1979 Motion by Commissioner of Patents and Trademarks to extend time for filing brief to April 27, 1979, filed.

April 24, 1979 Above motion granted

April 27, 1979 BRIEF FOR THE COMMISSIONER OF PATENTS AND TRADEMARKS, FILED.

May 9, 1979 Motion of appellants for extension of time for filing reply brief to May 21, 1979, filed.

May 11, 1979 Above motion granted.

May 11, 1979 REPLY BRIEF FOR APPELLANT FILED.

DATE	PROCEEDINGS AND ORDERS

June 6, 1979 Argued (Judge Cowen, sitting)

July 5, 1979 Reversed, Rich, Judge.

July 27, 1979 FINAL MANDATE ISSUED TO THE COMMISSIONER OF PATENTS AND TRADEMARKS

October 3, 1979 Extension of time granted by the Chief Justice of the Supreme Court in which to file a Petition for Writ of Certiorari to and including December 2, 1979. Solicitor General requested extension of time.

December 3, 1979 Petition for Writ of Certiorari filed in Supreme Court, No. 79-855.

March 17, 1980 Petition for Writ of Certiorari granted. (To be heard same day as 79-527 Diehr et al)

Patent Appeal No. 79-533

IN RE APPLICATION:

JOHN J. BRADLEY and BENJAMIN S. FRANKLIN

Serial No. 570,331

In the matter of the above referenced case which was heard before the United States Court of Custom and Patent Appeals, I was instructed to obtain the tape of the oral arguments from the court, listen to and transcribe the particular portion by the Solicitor.

I hereby certify that the following is a transcription of the portion requested:

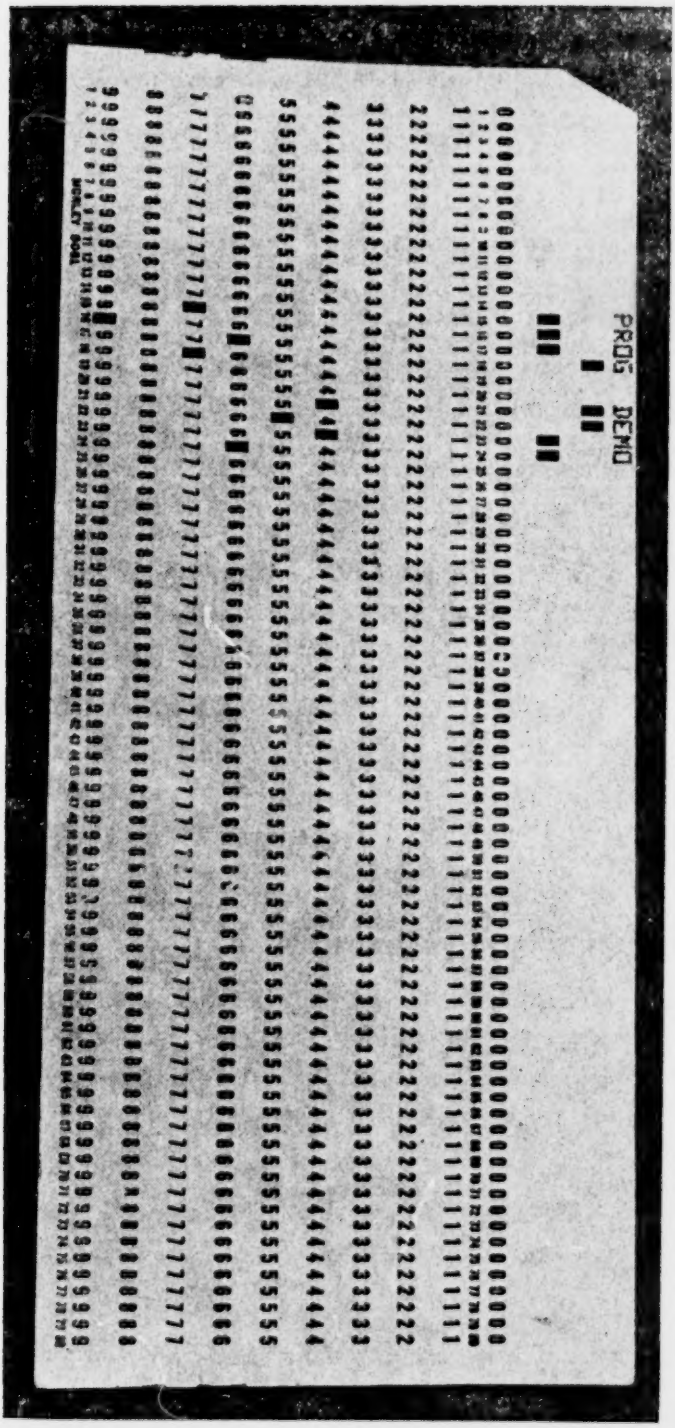
> "Do you agree with Applicant that he is claiming a machine where it starts out, 'In multiprogram computer system . . .'?"

Answer:

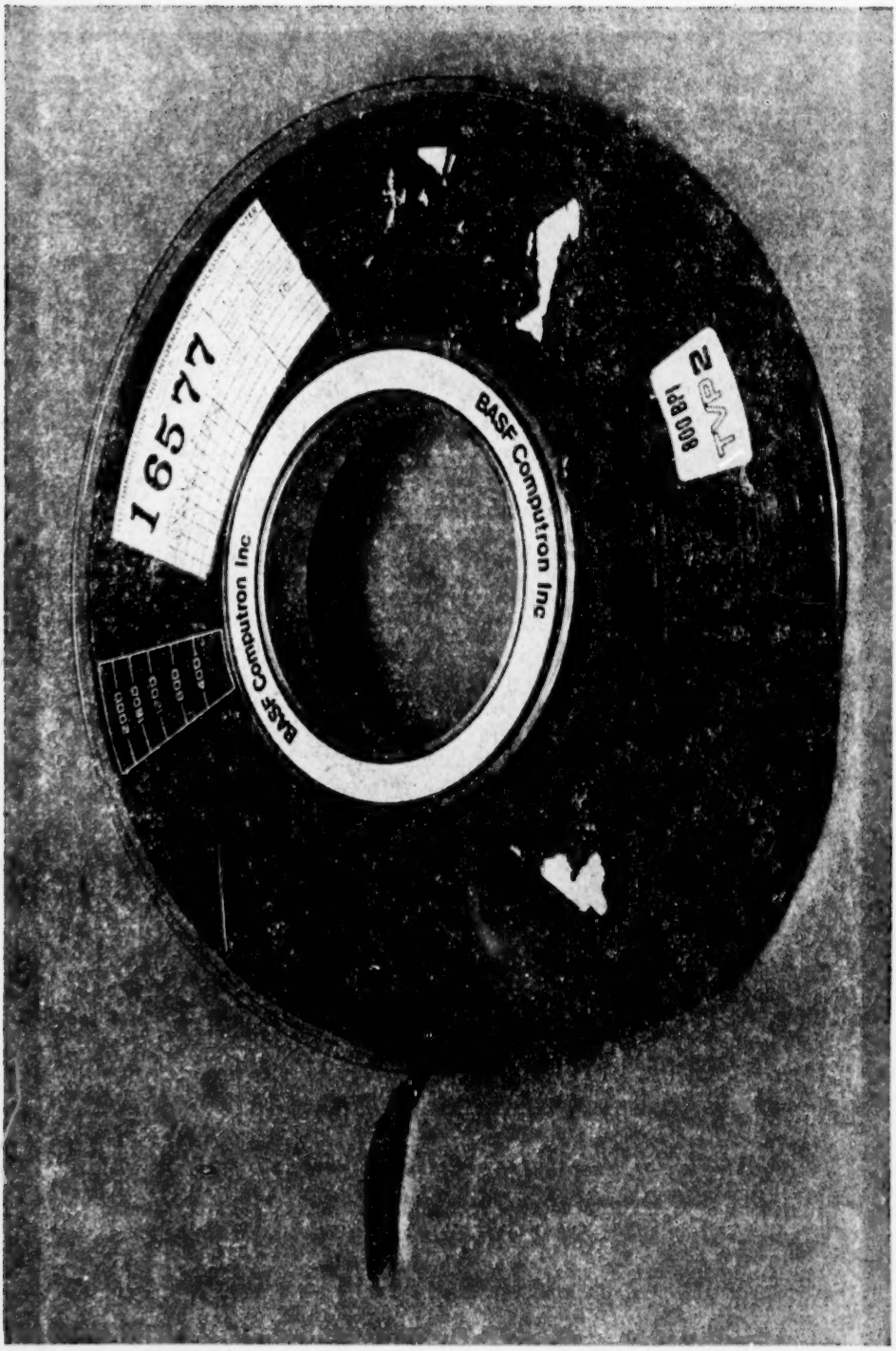
> "I suppose it is a machine, it is an apparatus of some sort, yes Your Honor."

Date: December 7, 1979

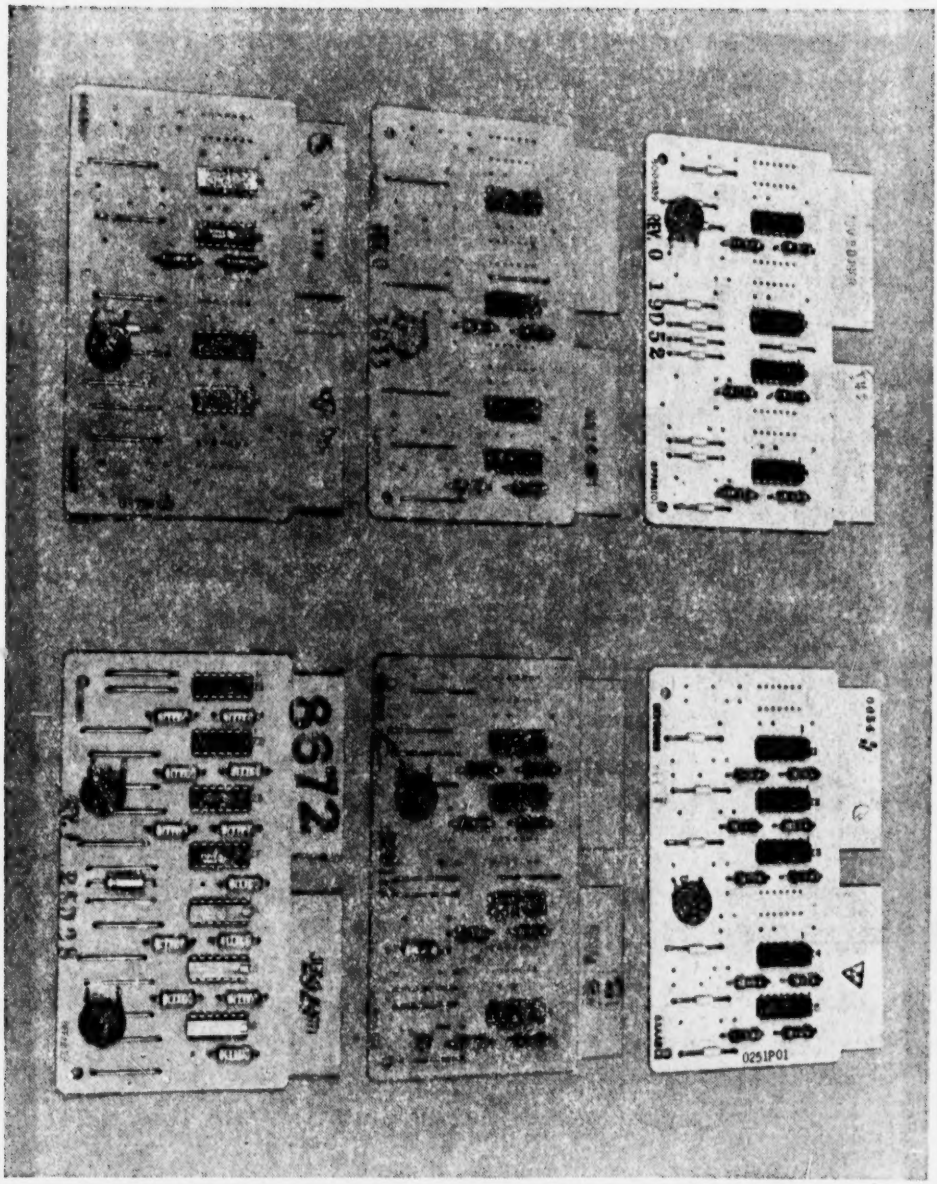
Signed Carolyn L. Leonard
2043 Crystal Plaza Drive
Arlington, Virginia 22202



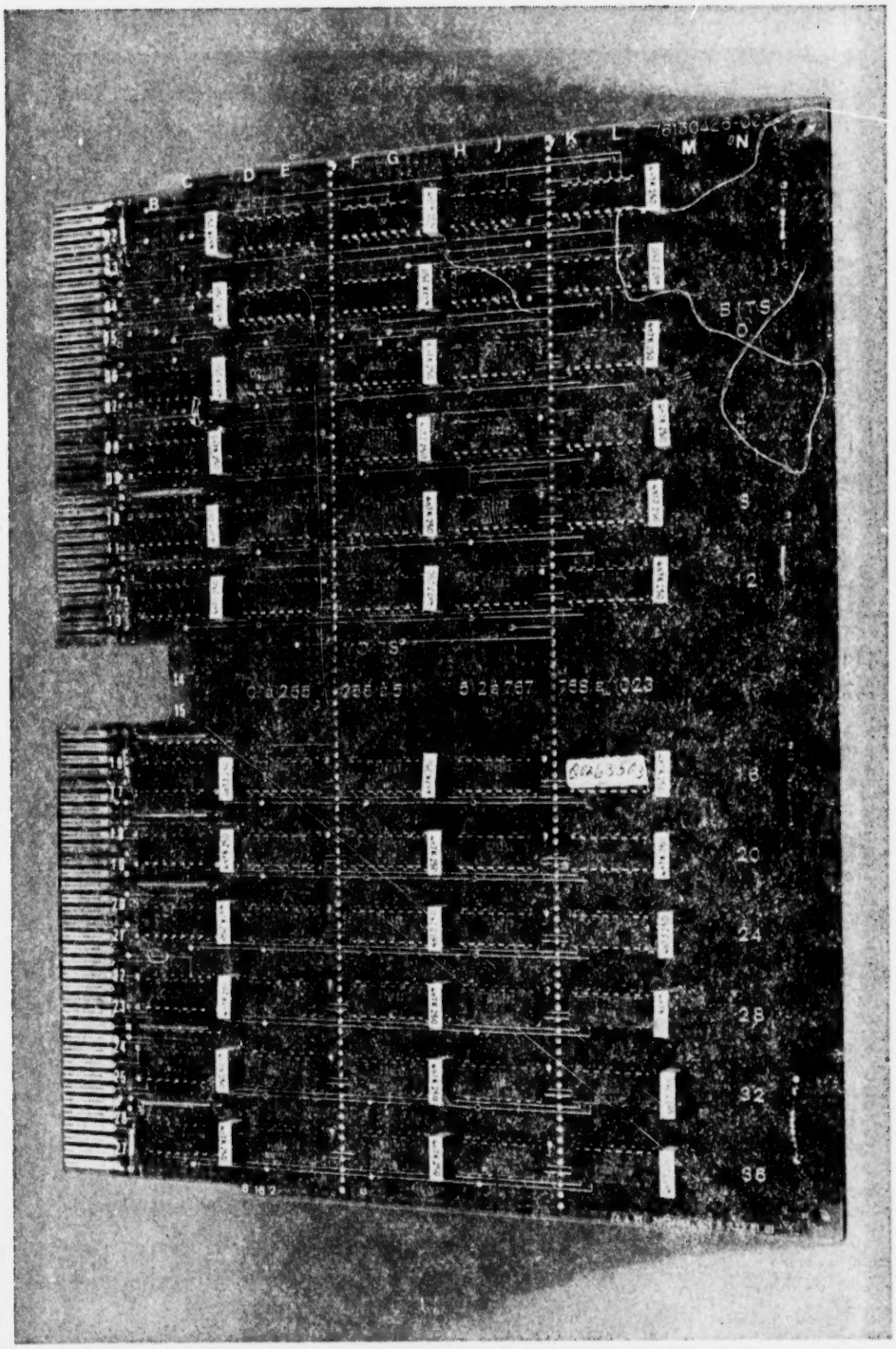
(a) Punch Cards of the type utilized by programmers in computer programming



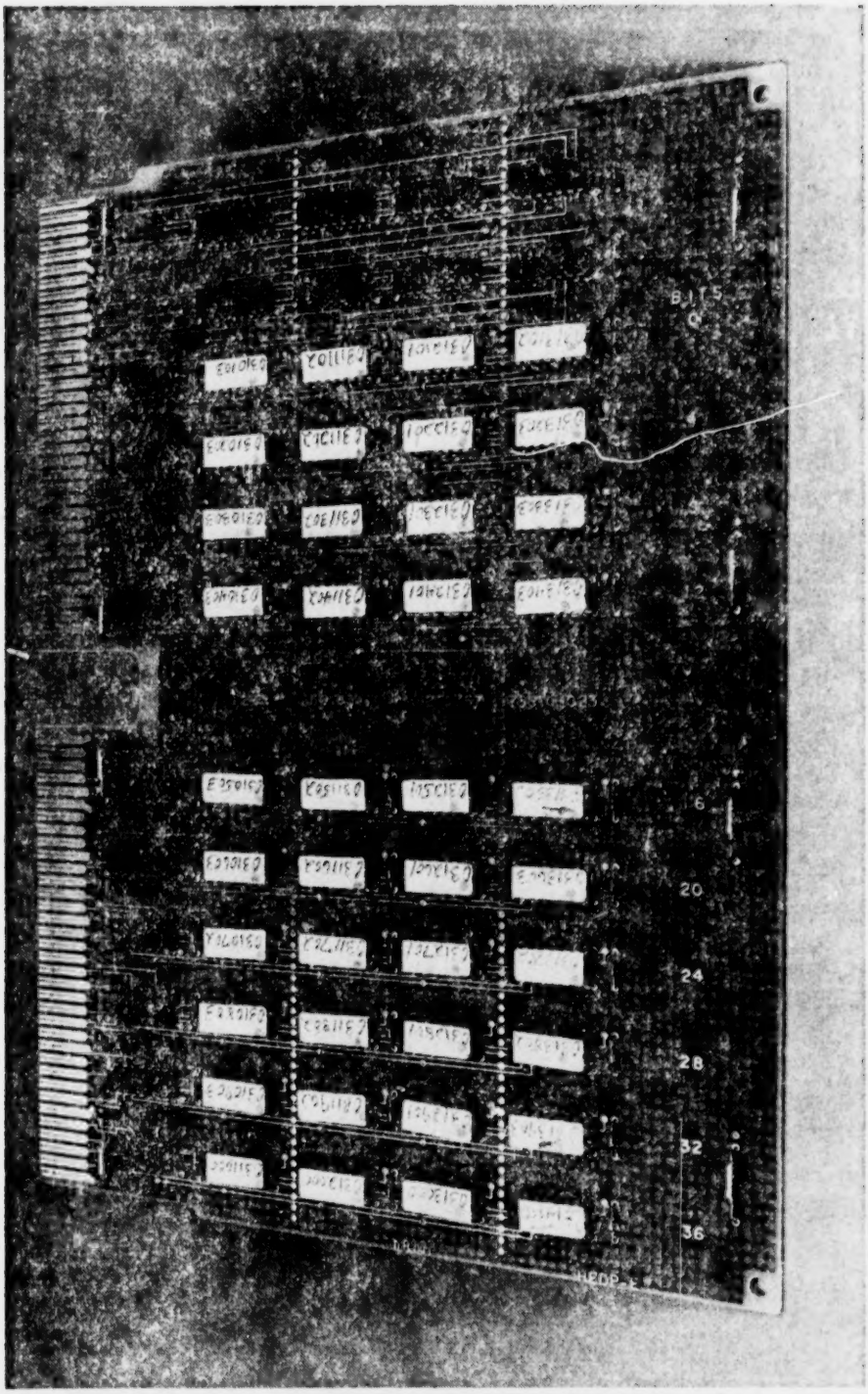
(b) Magnetic Tape of the type utilized by programmers in computer programming



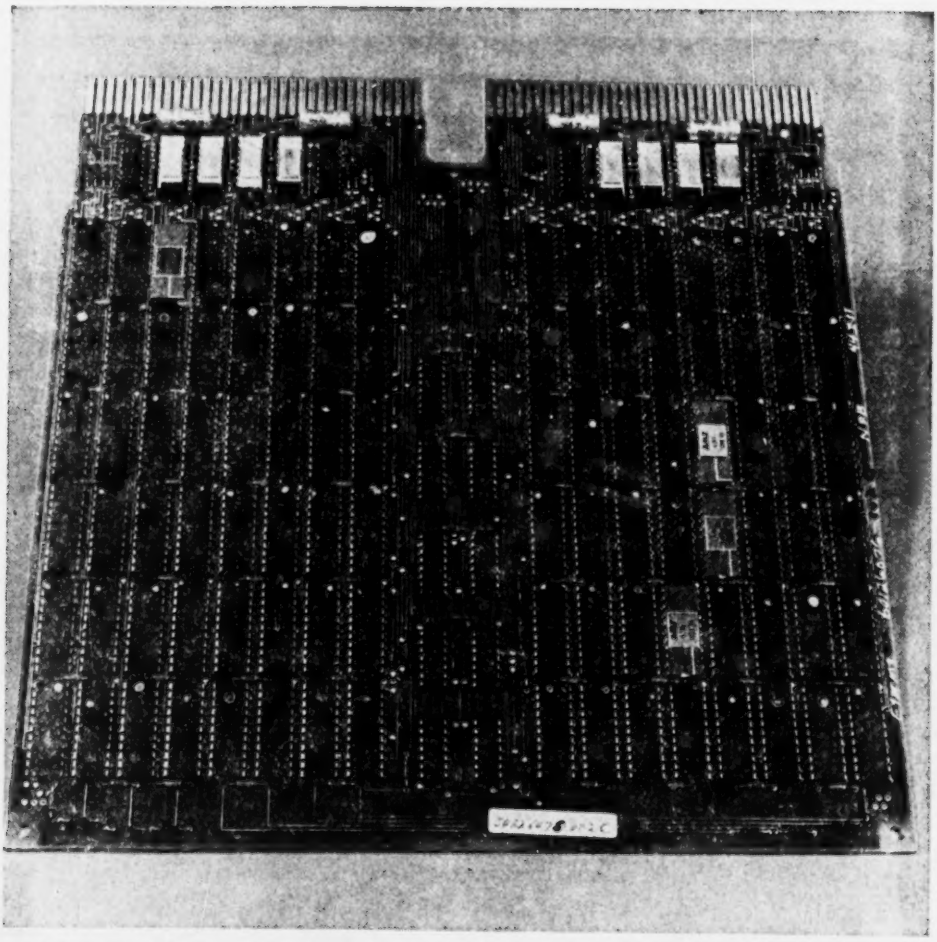
(c) A computer hardwired board of the type utilized by system designers to design new computer machines



(d) Two computer firmware boards of the type utilized by system designers to design new computer machines
Board (1)



(d) Two computer firmware boards of the type utilized by system designers to design new computer machines
Board (2)



(e) A MOS memory board of the type utilized for main memories in compter machines

UNITED STATES COURT OF CUSTOMS AND PATENT APPEALS

Appeal No. 79-533

IN THE MATTER OF THE APPLICATION OF
JOHN J. BRADLEY and BENJAMIN S. FRANKLIN

Serial No. 570,331

Decided: July 5, 1979

Before MARKEY, Chief Judge, RICH, BALDWIN, and MILLER, Associate Judges, and COWEN,* Senior Judge.

RICH, Judge.

This appeal is from the decision of the Patent and Trademark Office (PTO) Board of Appeals (board) affirming the rejection of claims 1-6, all of the claims in appellants' application serial No. 570,331, filed April 21, 1975, for "Switch System Base Mechanism," as being drawn to subject matter which is non-statutory under 35 USC 101. We reverse.

The Invention

Appellants' invention is in the field of computer technology. It does not relate to computer applications, i.e., any specific task that a computer is asked to perform, but rather to the internal operation of the computer and its ability to manage efficiently its operation in a multiprogrammed format. A multiprogrammed format is one in which the computer is capable of executing more than one program, and thus perform more than one applica-

* The Honorable Wilson Cowen, U.S. Court of Claims, sitting by designation.

tion at the same time, without the need to reprogram the computer for each task it must perform.

Specifically, the invention relates to altering or repositioning information in the computer's system base. The system base of a computer is a fixed area in main memory which acts as the root for all information structures in the computer. In high performance computer systems, it is very advantageous to store portions of the system base in "scratchpad" registers [1] located in the central processing unit (CPU) of the computer rather than in main memory. This greatly enhances the speed of operation of the computer because the access time (the time it takes to retrieve information from a given place) associated with the scratchpad registers is less than that associated with main memory. A problem arises, however, because a programmer may wish to change the positions or content of information in the system base which happens to be located in the scratchpad registers. These registers are "invisible" to the programmer since, unlike main memory, they cannot be accessed by software (computer programs). Prior art systems altered the system base information resident in the scratchpad registers by either reinitializing the system base (completely reloading the system base containing the new information), a process which consumes a considerable amount of time, or by using software which takes advantage of the model-dependent properties of the particular computer. The latter method has the undesirable effect of resorting to reliance on model-dependent software, which is unacceptable to some computer users.

Appellants' invention enables system base information to be altered without having to resort to these techniques and their accompanying drawbacks. They accomplish their result by employing a "firmware" module, consisting of hardware elements permanently programmed with a

---

[1] A scratchpad register, also known as a scratchpad memory, is a plurality of multibit storage locations, usually located in the central processing unit (CPU) of a computer, used for temporary storage of program information, operands, and calculation results for use by the computer's arithmetic and logic unit, and other information of a temporary nature.

microcode, which directs the data transfers, between the scratchpad registers and the system base located in main memory, which are necessary to effect the alteration.

Claim 1 is representative of the appealed claims:

> 1. In a multiprogramming computer system having a main memory, a central processing unit (CPU) coupled to said main memory, said (CPU) controlling the state of a plurality of groups of processes being in a running, ready, wait or suspended state, said computer system also having scratchpad registers being accessible to an operating system for controlling said multiprogramming computer system, a data structure for storing coded signals for communicating between said processes and said operating system, and said scratchpad registers, said data structure comprising:
>
> (a) first means in said data structure and communicating with said operating system for storing coded signals indicative of an address for a selected one of said processes;
>
> (b) second means in said first means for storing coded signals indicating priority of said selected one of said processes in relation to others of said processes for obtaining control of said CPU when ready;
>
> (c) third means in said data structure and communicating with said operating system, for storing coded signals indicative of an address for a selected one of said plurality of groups of processes, and,
>
> (d) fourth means coupled to said data structure and said scratchpad registers, for generating signals causing the changing of information in said data structure and said scratchpad registers.

The Rejection

The examiner rejected the appealed claims on the authority of *Gottschalk* v. *Benson*, 409 U.S. 63, 175 USPQ 673 (1972) (hereinafter *Benson*) before the Supreme

Court's decision in *Parker* v. *Flook*, 437 U.S. 584, 198 USPQ 193 (1978) (hereinafter *Flook*). In his final rejection, dated October 27, 1976, the examiner stated that the subject matter "deemed as the invention" is "a data structure" and then made the following analysis:

> The term "data structure" as recited in the claim is comprised of four entities described as "first means", "second means", "third means", and "fourth means." Whether these so called "means" are hardware structure which store coded signals for performing a particular function or whether they refer to coded signals representing computer control words per se, any life which the claim may have in view of that which is admittedly old in the art, resides in the "technique" (page 5, line 4) or "switch-system base instruction" (page 5, lines 13 and 14) for solving the problem of changing information of the system base which is located in scratch-pad registers which software cannot access (page 4, line 20). The invention then is embodied in the coded signals representing instructions to the computer, the running process word RPW (first means of claim), priority level indicator word PRI (second means), J Table word and G Table word (third means), and micro-instructions of Figures 15c [sic, 15b] and 15c as found in the central store 1301. The invention resides in a "data structure" or an algorithm designed to control the multiprogramming computer to solve the particular problem indicated.
>
> Under the ruling in *Gottschalk* v. *Benson*, 175 USPQ 673, the instant claims, depending upon a program implemented algorithm for patentability, are deemed nonstatutory subject matter.

Appellants requested reconsideration and argued that their claims are directed to data structures in hardware which are "specific new, novel and unobvious [in] arrangement." They asserted that by stating that the invention resided in a "technique," the examiner was clearly disregarding the claims and interpreting the invention strictly on the basis of what is found in the specification,

because no "technique" is *claimed*. They stated that even if a technique (i.e., process) were claimed, *Benson* does not render all such inventions nonstatutory, and that their invention does not involve a *mathematical* algorithm.[2]

In his Answer before the board, the examiner noted that all of the limitations found in claim 1 were old in the art and that the "claim is thus reciting prior art coupled with subject matter which the U.S. Supreme Court has found to be non-statutory in *Benson*." Appellants responded in their reply brief that it makes no difference whether individual elements are old in the art and that it is the elements in combination which define the invention as a whole.

The Board

The board rendered its decision on September 20, 1978, after the Supreme Court's decision in *Flook*. After incorrectly stating that none of the claims recited the term "firmware," [3] the board analyzed appellants' claims element by element, concluding that the only novel ar-

---

[2] In *Benson*, the Supreme Court was clearly limiting its discussion to mathematical algorithms:

> The patent sought is on a method of programming a general purpose digital computer to convert signals from binary coded decimal form into pure binary form. *A procedure for solving a given type of mathematical problem is known as an "algorithm." The procedures set forth in the present claims are of that kind; that is to say, they are a generalized formulation for programs to solve mathematical problems of converting one form of numerical representation to another.* [409 U.S. at 65, 175 USPQ at 674 (emphasis ours).]

It is not clear whether the examiner regarded appellants' invention as a mathematical algorithm, but, as will become evident, the board did so regard the invention.

[3] Claim 3, as well as dependent claims 4-6, explicitly recites firmware. Firmware is a term of art in the computer field and refers to microinstructions permanently embodied in hardware elements. For a further discussion, *see generally* Ross, *The Patentability of Computer "Firmware,"* 59 JPOS 731 (1977). We need not and do not decide at this time whether firmware *per se* is statutory under 35 USC 101 because the invention *as a whole* is not directed thereto. See our further discussion, infra.

rangement of the recited structures resided in the microprogramming, "which together with its attendant memory hardware appears to constitute firmware."

Apparently on the basis of *Flook*, the board affirmed the rejection because it was of the opinion that the appealed claims are directed to a method of calculation or mathematical algorithm. The board found the claims similar to the claims at issue in *In re Waldbaum*, 559 F.2d 611, 194 USPQ 465 (CCPA 1977), which included language characterized by the board as "obviously related to calculating and mathematical problem solving."[4] Although the claims here at issue do not contain similar mathematical language, the board said that this "does not make the functions attendant the 'means' of appellants' claims any less mathematical or less related to an algorithm within the meaning assigned that term by the USSC in *Benson*."

To support its conclusion that the appealed claims are mathematical in nature, the board relied on a statement in the specification to the effect that all of the data in the computer are in binary form, but may be interpreted as binary coded decimal, decimal or alphanumeric. We reproduce the board's reasoning in full:

> Since digital computers normally operate in some numerical radix, binary, binary coded decimal, or the like, we consider the operation of appellants' claimed invention to be mathematical.
>
> Every operation performed in appellants' invention as claimed involves the accommodation of data and instructions to the size of the registers in memory, and to the positional assignment to the registers in memory by the use of some numerical measure or quantity effected by way of electrical signals. In

---

[4] Waldbaum claim 1 reads in pertinent part:

> a method * * * to *count the number* of busy lines * * * *comparing means* * * * *to derive the number of l's* in said data word * * *. [Emphasis ours.]

It is clear that this claim recites a mathematical algorithm. It solves a mathematical problem, to wit, *counting* a number of busy lines in a telephone system.

> whatever form the instructions employed in appellants' invention are characterized, numerical or otherwise, we think it is accurate to say that the operation of appellants' structure is mathematical and that the instructions constitute a procedure which is algorithmic in character, to the same degree as that of the Waldbaum structure and that satisfactory operation of the apparatus claimed represents the successful solution of a mathematical problem falling within the definition of algorithm supplied in *Benson* and reiterated in *Flook*.
>
> Here the method of calculation is to be used in the internal operation of the computer system and the claims recite such an end use by means of the functions recited for the various means. A claim for an improved method of calculation, even when tied to a specific end use, is unpatentable subject matter under 35 USC 101, *Flook*, *supra*.

In summary, the board stated that the claims are drawn to apparatus in form only, and couple the apparatus (which it asserts is old in the art) "with subject matter, namely, programming, which is nonstatutory under the *Benson*, *Christensen*,[5] *Waldbaum* and *Flook* cases * * *."

OPINION

The examiner's basis for the rejection is grounded on the erroneous interpretation of the Supreme Court's decision and opinion in *Benson*, namely, that all computer program or program-related inventions are nonstatutory under § 101. Both the Supreme Court[6] and this

---

[5] *In re Christensen*, 478 F.2d 1392, 178 USPQ 35 (CCPA 1973).

[6] In *Benson*, the Court stated, 409 U.S. at 71, 175 USPQ at 676 (emphasis ours):

> It is said that this decision precludes a patent for any program servicing a computer. *We do not so hold.*

In *Flook*, the Court stated, 437 U.S. at 595 n.18, 198 USPQ at 199 n.18:

> Very simply, our holding today is that a claim for an improved method of calculation, even when tied to a specific end use, is unpatentable subject matter under § 101.

court[7] have thoroughly repudiated this view. Our decision, therefore, is based solely on the analysis made by the board.

The board said that claims do not *directly* recite a mathematical formula, algorithm, or method of calculation, but, nevertheless, held the claims to be mathematical in nature. As appears from the quoted portion of the board opinion, the board regarded the fact that digital computers operate in some number radix as conclusive on the issue of whether the appealed claims recite a mathematical algorithm in the *Benson* and *Flook* sense. The board did not, however, direct attention to any specific formula it thought is utilized, or to what, if anything, the mathematical calculations alleged to be present in the claims are directed.

We do not agree with the board. We are constrained to reject its reasoning. Such reasoning leads to the conclusion that any computer-related invention must be regarded as mathematical in nature, a conclusion which is not compelled by either *Benson* or *Flook*.

The board's analysis confuses *what* the computer does with *how* it is done. It is of course true that a modern digital computer manipulates data, usually in binary form, by performing mathematical operations, such as addition, substraction, multiplication, division, or bit shifting, on the data. But this is only *how* the computer does what it does. Of importance is the significance of the data and their manipulation in the real world, i.e., *what* the computer is doing. It may represent the solution of the Pythagorean theorem, or a complex vector equation describing the behavior of a rocket in flight, in which case the computer is performing a mathematical algorithm and solving an equation. This is what was in-

---

[7] *See In re Gelnovatch*, 595 F.2d 32, 36-37, 201 USPQ 136, 141 (CCPA 1979); *In re Johnson*, 589 F.2d 1070, 1075, 200 USPQ 199, 205 (CCPA 1978); *In re Sarkar*, 588 F.2d 1330, 1333, 200 USPQ 132, 137 (CCPA 1978); *In re Freeman*, 573 F.2d 1237, 1244, 197 USPQ 464, 470 (CCPA 1978); *In re de Castelet*, 562 F.2d 1236, 1240, 195 USPQ 439, 443 (CCPA 1977); *In re Chatfield*, 545 F.2d 152, 155-56, 191 USPQ 730, 733-34 (CCPA 1976), *cert. denied*, 434 U.S. 875, 195 USPQ 465 (1977).

volved in *Benson* and *Flook*. On the other hand, it may be that the data and the manipulations performed thereon by the computer, when viewed on the human level, represent the contents of a page of the Milwaukee telephone directory, or the text of a court opinion retrieved by a computerized law service. Such information is utterly devoid of mathematical significance. Thus, the board's analysis does nothing but provide a quick and automatic negative answer to the § 101 question simply because a computer program is involved.

Appellants have continuously insisted that they are claiming a new and unobvious combination of *hardware elements*, i.e., a new machine or apparatus.[8] The issues of novelty and unobviousness[9] are not before us, but we agree with appellants insofar as they characterize their invention as a machine or apparatus. The board likewise seems to agree on this point. In its opinion, it identifies all of the "means" of appellants' claim 1 as structural hardware elements, such as registers, portions of main memory and control store,[10] and other computer components. Thus, the claim falls literally within the boundaries of § 101.

Appellants have characterized their combination of hardware elements as a mechanism which enables the computer to alter information in its system base in a manner not previously possible. They are in no way claiming the altered information; in fact, the particular information acted upon by appellants' invention is irrelevant to the operation of the invention itself. We see no difference in this regard, with respect to being within

---

[8] 35 USC 101 states:

> Whoever invents or discovers any new and useful process, *machine*, manufacture, or composition of matter, *or any new and useful improvement thereof*, may obtain a patent therefor, subject to the conditions and requirements of this title. [Emphasis ours.]

[9] 35 USC 102, 103.

[10] The control store of a computer is a plurality of multibit storage locations (memory) containing microinstructions, which, when decoded by appropriate circuitry, provide control signals which cause specific operations to take place in the computer's CPU.

§ 101, between appellants' claimed invention and a strictly mechanical adding machine, which is certainly statutory if claimed in a manner which does not embrace any particular calculation that the machine is capable of making.

The PTO regards as significant the fact that firmware is involved in the present invention. In a sense, firmware may be likened to software (computer programs); it is information which has been embodied into hardware by, for example, destroying fusible links in a read only memory (ROM) array. In appellants' invention, the information contained within the firmware, which is located in the control store of the computer, directs the desired information transfers within the system base.

If appellants were claiming the information embodied in the firmware or the firmware itself, per se, a different case would be presented. We express no opinion on the statutory nature of such an invention, a question not before us. Appellants are claiming a combination of hardware elements, one of which happens to be a portion of the computer's control store microprogrammed in a particular manner. It is this subject matter with which we must deal.

From our reading of appellants' specification, their claimed "data structure" is merely that which results from the arrangement of the recited hardware elements in the claimed manner. It is the result of certain structural "means" performing certain recited functions as explicitly sanctioned by 35 USC 112, sixth paragraph. We disapprove the board's distillation of appellants' claim down to the information contained in the firmware in order to hold it nonstatutory. The invention is not claimed in that manner, and, in this case, we see no reason to view the claim format as a subterfuge for masking the presence of an essentially nonstatutory invention.

Even though the claimed invention is a machine, we must nevertheless determine whether the claim recites a mathematical algorithm, and, if so, whether it preempts the use of the algorithm. *In re Noll*, 545 F.2d 141, 148, 191 USPQ 721, 726 (CCPA 1976). In doing so, we apply the two-step test in *In re Freeman*, supra note 7.

When we examine appellants' invention as a whole under the first step of this test, including the information microprogrammed into the firmware element as depicted in Figs. 14(a-i) and 15(b-c), we fail to detect the presence of any mathematical algorithm. In altering information in the system base as desired, certain "calculations" are made, such as determining whether a given quantity is equal to zero, or, as noted by the solicitor, multiplying an address in memory by sixteen to arrive at another address. However, it certainly cannot be said that comparing with zero or multiplying by sixteen is preempted by appellants' claims. Furthermore, the presence of these calculations does not transform the invention as a whole into a method of calculation. *Cf. In re Gelnovatch*, supra note 7. There is no solution of an equation, such as the new alarm limit in *Flook*, or the equivalent pure binary number, as in *Benson*, present in the computer after the task has been completed. *See In re Chatfield*, supra note 7.

In summary, we have examined the claims thoroughly and we do not find any mathematical formula or mathematical method of calculation, improved or otherwise, which is either claimed as such or essential to the vitality of the claims. We find that the invention is a combination of tangible hardware elements—a machine—including some hardware elements which contain microprogrammed information termed "firmware." We do not find the invention to be nonstatutory under the authority of *Benson* or *Flook*, or under the authority of our own cases, such as those cited at note 7 ante. Therefore, the decision of the board is *reversed.*

REVERSED

UNITED STATES COURT OF CUSTOMS
AND PATENT APPEALS

MANDATE

No. 79-533

IN THE MATTER OF THE APPLICATION

OF

JOHN J. BRADLEY and BENJAMIN S. FRANKLIN

Serial No. 570,331

ON APPEAL from the Board of Appeals

This CAUSE having been heard and considered, it is

ORDERED and ADJUDGED: Reversed

Dated July 5, 1979

A True Copy
GEORGE E. HUTCHINSON,
Test:
Clerk, United States Court of
Customs and Patent Appeals

Certified this 27th day of July 1979

/s/ George E. Hutchinson
GEORGE E. HUTCHINSON

IN THE UNITED STATES COURT OF
CUSTOMS AND PATENT APPEALS

Patent Appeal Docket No. 79-533

IN RE APPLICATION OF
JOHN J. BRADLEY and BENJAMIN S. FRANKLIN

APPEAL FROM BOARD OF APPEALS

Serial Number 570,331

PRAECIPE FOR TRANSCRIPT OF RECORD, NOVEMBER 17, 1978

November 13, 1978

Commissioner of Patents & Trademarks
Washington
District of Columbia 20231

Dear Sir:

Request is hereby made that a transcript of the record be prepared and filed in the United States Court of Customs and Patent Appeals for use in connection with the appeal to the United States Court of Customs and Patent Appeals from the decision of the Patent Office Board of Appeals rejecting the claims of the above-identified patent application, the said transcript to include the following:

(1) This praecipe for transcript of record.
(Serial Number 570,331, filed April 21, 1975)

(2) Application Serial No. 570,331 (including petition specification, drawings and declaration or oath as filed, but not including the claims).

(3) Letter of the Examiner (First Office Action) March 12, 1976 (5 pages).

(4) Request for Extension of Time, May 18, 1976, and grant thereof.

(5) Amendment A, August 10, 1976.

(6) Letter of the Examiner (Second Office Action) October 27, 1976.

(7) Amendment Under Rule 116, December 28, 1976.

(8) Advisory Action, February 1, 1977.

(9) Notice of Appeal to Board of Appeals dated October 27, 1976.

(10) Brief for Appellant.

(11) Examiner's Answer Mailed August 4, 1977.

(12) Appellants' Reply Brief to Examiner's Answer.

(13) Notice of Hearing, Mailed December 13, 1977.

(14) Oral Hearing Confirmation dated January 16, 1978.

(15) Transcript of Oral Hearing (If Available).

(16) The claims on appeal (a copy of such claims are on pages 6-8 of the Brief for Appellant).

(17) Decision of Board of Appeals mailed September 20, 1978.

(18) Notice of Appeal to the Court of Customs and Patent Appeals and Reason for Appeal dated

It is requested that the Patent Office telephone the undersigned when the certified transcript of the record is ready so that the record can be picked up and delivered to the Clerk of the Court of Customs and Patent Appeals.

It is further requested that costs for preparation of the transcript be charged to Honeywell Information Systems Inc. Deposit Order Account No. 08-2732.

Respectfully submitted,

By: /s/ Nicholas Prasinos
NICHOLAS PRASINOS
Attorney for Applicants
Registration No. 24,791

APPLICATION OF JOHN J. BRADLEY AND BENJAMIN S. FRANKLIN, FILED APRIL 21, 1975, SERIAL NUMBER 570,331, FOR SWITCH SYSTEM BASE MECHANISM.

ABSTRACT OF THE DISCLOSURE

A model-independent instruction in firmware is provided to change, upon execution, information in the system base which is located in both main memory and internal scratch pad registers.

RELATED APPLICATIONS

The following applications are incorporated by reference to the instant application.

1. "Buffer Store" invented by J. L. Curley, T. J. Donahue, W. A. Martland, and B. S. Franklin, filed on October 5, 1972 having Serial Number 295,301 and assigned to the same assignee named herein.

2. "Variable Masking for Segmented Memory" invented by Wallace A. Martland and John L. Curley, filed on October 5, 1972 having Serial Number 295,303 and assigned to the same assignee named herein.

3. "Override Hardware for Main Store Sequencer" invented by Thomas J. Donahue, filed on October 5, 1972 having Serial Number 295,418 and assigned to the same assignee named herein.

4. "Main Memory Sequencer" invented by T. J. Donahue, J. L. Curley, B. S. Franklin, W. A. Martland, and L. V. Cornaro, filed on October 5, 1972 having Serial Number 295,331 and assigned to the same assignee named herein.

5. "Main Memory Reconfiguration" invented by J. L. Curley, B. S. Franklin, W. A. Martland, T. J. Donahue and L. V. Cornaro filed on October 5, 1972 having Serial Number 295,417 and assigned to the same assignee named herein.

6. "Ring Checking Hardware" invented by ———— filed on the same date as the instant invention, having Serial Number ———— and assigned to the same assignee named herein.

7. "Segmented Address Development" invented by ——————, filed on ———— and having Serial Number ———— and assigned to the same assignee named herein.

8. "Protection of Information in a Multiprogram Multiprocessor Computer System" invented by ————, filed on ———— and having Serial No. ————, and assigned to the same assignee named herein.

9. "Ring Crossing Mechanism" invented by ————, filed on ————, and having Serial Number ———— and assigned to the same assignee named herein.

10. "Procedure Calls and Stack Operations", invented by ————, filed on ————, and having Serial Number ———— and assigned to the same assignee named herein.

11. "Process Synchronization Utilizing Semaphores" invented by ————, filed on ————, and having Serial Number ————.

12. "Arrangement for Automatically Managing, Scheduling and Allocating System Resources" invented by ——————, filed on ———— and having Serial Number ———— and assigned to the same assignee named herein.

13. "Data Processing Memory Management System" invented by ————, filed on ———— and having Serial Number ———— and assigned to the same assignee named herein.

14. "Arrangement for Initializing a Computer System Controlled by Reloadable Site-Dependent Information Structures", invented by ————, filed on ———— and having Serial Number ———— and assigned to the same assignee named herein.

15. "System Base", invented by ————, filed on ————, and having Serial Number ———— and assigned to the same assignee named herein.

16. "Absolutization During Initialization", invented by ————, filed on ————, and having Serial Number ———— and assigned to the same assignee named herein.

BACKGROUND

Field of the Invention:

This invention relates generally to computer systems and more particularly to a system and method for communicating between hardware and software in a multiprogramming/multiprocessing environment.

Multiprogramming/multiprocessing computer systems require the use of a communications area in main storage for communications between hardware and software. (See the application entitled "System Base" invented by ———— filed on ———— having Serial Number ———— and assigned to the instant assignee, and which application has been incorporated by reference herein). The system base is generally located at a fixed location in main memory, although this location may be changed or relocated through the use of a device such as a boundary address register (BAR). (See invented by ————, filed on ————, having Serial Number ———— and assigned to the instant assignee, and further incorporated by reference in the instant application).

High performance computing systems find it necessary to store portions of the system base in internal scratchpad registers instead of main memory, and hence inaccessible and invisible to software running in the system. Accordingly, a problem results when software (i.e. the programmer) wants to change positions or change information of the system base which happens to be located in these internal central processor scratchpad registers which software cannot access.

Prior Art:

The problem has been solved in one of two ways. The first method is to simply re-initialize the entire system, loading the system base with different contents as a result of a new initialization process. This technique is extremely time-consuming and quite undesirable.

A second technique for solving the problem is to have the software employ special model-dependent properties

of the particular machine on which it was running, to change the system base. This results in the very undesirable property of completely model-dependent software—a feature which certainly does not serve to endear a computer manufacturer to a customer.

What is needed therefore is a technique of changing information in the system base both in main memory and internal scratchpad registers making the system base in scratchpad memory accessible to software but not visible to software i.e. cannot be read by the software.

OBJECTS OF THE INVENTION

It is a primary object of the invention to provide an improved system and method for changing information in the system base of a computer system.

It is another object of the invention to provide a switch-system base instruction in firmware which changes, upon execution, information in the system base.

It is still another object of the invention to provide a system and method for changing information in the system base which is located in main memory and scratchpad registers.

Yet another object of the invention is to provide an improved switch system base mechanism.

These and other objects of the invention will become apparent from the description of a preferred embodiment of the invention when read in conjunction with the drawings contained herewith.

SUMMARY OF THE INVENTION

A model-independent instruction in firmware is provided to change, upon its execution, information in the system base which is located in both main memory and internal scratchpad registers.

The system base located at a location specified by the boundary address register BAR in main memory, is copied into a new location in main memory with predetermined portions of its information changed. This is readily accomplished by the operating system. (See ap-

plication incorporated by reference into the application herein). The switch system base instruction then transfers the information which is in the system base at the new location into the system base at the location specified by BAR and also that portion of the system base located in scratchpad memory.

BRIEF DESCRIPTION OF THE DRAWINGS

The novel features which are characteristic of the invention are set forth with particularity in the appended claims. The invention itself, however, both as to organization and operation together with further objects and advantages thereof may best be understood by references to the following description taken in conjunction with the drawings in which:

Figure 1 is a block diagram of a multiprogramming system utilizing the invention.

Figure 2 is a schematic representation of various hardware structures utilized by the invention.

Figure 3 is a legend of terms used for reserved areas of storage in registers depicted in Figure 2.

Figure 4 is a schematic diagram of a process control block.

Figure 5 is a schematic diagram of a system for addressing a process control block.

Figure 6 is a schematic diagram of the system base of the invention.

Figure 7A and 7B are a schematic representation of a stack segment and a stack frame respectively.

Figure 8 is a schematic diagram of a system for addressing G-segments and in particular the queue of processes in the GO segment.

Figure 9 is an exploded schematic diagram of a GO segment illustrating queue of processes and process linking.

Figures 10a through 10L are block diagrams of structures in the PCB.

Figures 11a through 11r are block diagrams of structures in the system base.

Figure 12 is a schematic diagram of addressing schemes of user and system segments utilizing the system base and PCB structures.

Figure 13 is a schematic diagram of the control unit.

Figures 14a through 14i are flow diagrams of the dispatcher unit in firmware.

Figure 15a is a structure of the switch system base instruction.

Figures 15b-15c are flow diagrams of the switch system base instruction in firmware.

DESCRIPTION OF A PREFERRED EMBODIMENT

GENERAL DISCUSSION

The invention operates typically in the hardware system environment hereinafter described coordinated by a hardware/firmware/software operating system. Referring to Figure 1 the subsystems are the processor subsystem 101, the storage subsystem 102, and one or more —up to 32—peripheral subsystems 103. The processor subsystem contains a central processing unit (CPU) 104 and up to four input/output control units (IOC) 105. Each peripheral subsystem consists of a peripheral control unit (PCU) 106, a number of device adapters (DA) 107, and up to 256 peripheral i/o devices 108. The storage subsystem 102 consists of one to four semiconductor memory modules of 32 to 512 kilobytes each.

I. PROCESSOR SUBSYSTEM

In the processor subsystem 101, the CPU 104 performs the basic processing operations for the system, and interfaces with memory 102. The IOC 105 controls all information exchanges between the storage subsystem 102 and peripheral devices 106.

A. CENTRAL PROCESSING UNIT

The CPU includes a main memory synchronizer 109, a buffer store 110, various elements that comprise the

computational unit 111, and optional emulation facilities 112. The main memory synchronizer 109 resolves conflicts for the use of main memory among the computational unit 111, the buffer store 110, and the IOC 109. Conflicts are resolved on a priority basis: the IOC has the highest priority followed by memory writes (from the computational unit) and then memory reads (into the buffer store). The main CPU also includes the address control unit ACU 131 which controls main memory addressing and the associative memory AS 132 used to store most recently used addresses of main memory. The buffer store 110 is a small high-speed buffer memory that reproduces a selected region of main memory and interfaces with the computational unit to decrease average memory access time. During each memory read, both the buffer store and main memory are accessed. If the information to be fetched is already in the buffer store, the main memory read is terminated and the information fetched from the buffer store. Otherwise the main memory 102 is read. Every time this is done, the CPU 104 fetches 32 bytes that contains the desired information. This information remains in the buffer store for future memory references. Since the buffer store is transparent to software, the program controlling the computer at any given moment cannot determine whether the information it is processing has been fetched from the buffer store or from main memory.

The computational unit 111 performs all data processing and address generation within the CPU. A typical control store 130 within the computational unit (see a book entitled Microprogramming: Principles and Practices, Samir S. Husson, Prentice Hall, Inc.) contains firmware which initializes the system, controls the CPU 104 and IOC 105, and decodes an instruction set (not shown). Optionally the control store may provide scientific instructions, test routines, emulation packages, or special purpose features which extend the capabilities of the processor subsystem.

As an option, the CPU provides emulation of systems other than the instant system. Emulators 112 are components of firmware, software, and in some instances hardware.

B. INPUT-OUTPUT CONTROL UNIT

The IOC 105 portion of the processor subsystem provides a data path between any peripheral subsystem 103 and the storage subsystem 102. This path allows for the initiation of peripheral commands and controls the resulting data transfers. An IOC can typically handle up to 32 channel control units (not shown).

C. PERIPHERAL SUBSYSTEMS

In a peripheral subsystem 103 on Figure 1 the PCU 106 is a stand-alone microprogramming processor that relieves the load on the CPU 104 by controlling the i/o devices 108 during i/o operations. The PCU does this by executing instructions contained in a channel program. This program results in arithmetic, logical, transfer, shift, and branch operations being performed in the PCU. There are several kinds of PCU's according to the kind of device each controls: i.e. unit record, mass (disk) storage, magnetic tape, communications, etc.

Device adapters 107 mediate between every PCU and the devices it controls. Each contains the dedicated firmware and logic necessary to implement communication with a particular type of device. Depending on the type, a DA 107 controls one or several devices.

The major functions performed by a peripheral subsystem 103 are as follows:

1. Transforming CPU instructions into a series of commands acceptable to the appropriate peripheral device.
2. Packing and unpacking data in the form needed by the CPU or the appropriate peripheral device.
3. Keeping the CPU informed of the status of the subsystem and of the devices under its control.
4. Independently initiating and processing error and recovery procedures.
5. Allowing on-line diagnosis of a device without disturbing the device-sharing capabilities of the associated peripheral processor.

A PCU resolves conflicts for main memory between devices attached to it; however, the IOC resolves conflicts between PCU's.

D. STORAGE SUBSYSTEM

Each memory module 1-4 is 4 or 8 bytes wide. The number of modules, their size, and the data path width may vary according to size of computer. Memory modules are four-way interleaved in such a way that the four modules are accessed sequentially (module 1 contains the first 8 bytes, module 2 contains the second 8 bytes, etc.). Interleaving decreases the number of conflicts for access to main memory and thereby decreases the average memory access time. Memory is reconfigurable in case of failure; i.e., blocks of memory within a module may be removed without destroying contiguous addressing.

Main memory 102 consists of a capacitive storage medium in the form of metal oxide semiconductor (MOS) chips. This medium operates on the refresh principle to maintain information. Each memory location is typically refreshed at least once every 2 milliseconds; the design ensures that few conflicts occur between refresh timing and memory accesses. (In cases of conflict, refreshing takes precedence).

An area at the beginning of main memory is reserved for hardware and firmware. The upper limit of this area is defined by the content of a boundary address register (BAR—to be later described) which is visible to the system software. The BAR content is set at system initialization time. The memory area below the address specified in the BAR can contain IOC tables which define the configuration of the peripheral subsystems, firmware to control the CPU, or microprograms and tables for emulation. The size of the area below the address specified in the BAR depends on the system configuration. Whether microprograms are in main memory or control store depends on the system configuration and the applications run on the system.

II. BASIC MACHINE STRUCTURES

There are typically three basic data structures utilized in this hardware: data formats, software visible registers, and the instruction formats.

A. DATA FORMATS

Information is transferred between memory and the CPU in multiples of 8 parallel bits. Each 8-bit unit of information is called a byte. Parity or error correction data is also transferred with data but cannot be affected by software. Therefore, in this patent specification the term data excludes the associated parity or error correction data.

B. BYTES

Bits within a byte are numbered 0 through 7 from left to right. Bytes are processed separately or in groups. Two bytes constitute a halfword, 4 bytes a word, 8 bytes a doubleword, and 16 bytes a quadword. These are the basic formats for all data, including instructions.

C. DATA REPRESENTATION

All data are in binary form, but may be interpreted as binary, decimal, or alphanumeric. Data bits are interpreted in groups of four, as binary coded decimal data; eight as alphanumeric, or 16 to 64 as binary digits. The latter are interpreted as signed, fixed, or floating-piont numbers in binary notation. Any number of contiguous bits up to a doubleword may also be manipulated as a string. The alphanumeric character set is represented in EBCDIC. ASCII is supported as an alternate exchange code.

D. BYTE ADDRESSES

Byte locations in main memory are consecutively numbered starting with zero; each number is the address of the byte. A group of consecutive bytes is said to be

halfword-, word-, doubleword-, or quadword-aligned, if the address of the left byte in a group is a multiple of 2, 4, 8, or 16, respectively. Whenever a halfword, word, doubleword, or quadword is so aligned, that unit can be fetched from that address. The location of data in main memory is specified by a data descriptor which is accessed indirectly during address development. (See Patent Application No. ———— filed ———— entitled Segmented Address Development and assigned to the same assignee as the instant application).

E. VISIBLE REGISTERS

There are 33 user-visible registers in the CPU 104 Figure 1 whose contents collectively define the state of the CPU. There are four types: (See Figure 2).

1. general registers
2. base registers
3. scientific registers (optional)
4. miscellaneous registers

F. GENERAL REGISTERS

General registers (GR) 201 are used to manipulate fixed-point binary numbers and bit strings. There are typically sixteen 32-bit general registers in the CPU 104—GR0 through GR15. General register GR8 through GR15 are also usable as index registers. When used as index registers, they are herein called X0 through X7: Indexing is performed using the 32-bit two's component integer contained in a register.

G. BASE REGISTERS

Base registers (BR) have the same format as instruction counters IC and stack registers 202-203. Base registers are used during address computation to define a part of memory. There are typically eight 32-bit base registers, BR0 through BR7.

H. SCIENTIFIC REGISTERS

Scientific registers (SR) are optional equipment for computation with floating-point binary numbers. There are typically four 8-byte scientific registers which are referred to as SR0 through SR3. Scientific registers have the format 204-205 of Figure 2.

I. MISCELLANEOUS REGISTERS

There are five other registers:

- instruction counter—having format 202-203;
- status register—having format 207;
- stack register (called the T register);
- boundary address register—having format 202-203; and
- hardware control mask register—having format 208.

The instruction counter (IC) is a 32-bit register that contains the address of the instruction being executed. The status register (STR) 207 is an 8-bit register that records facts about the procedure currently being executed, for example, whether an underflow was caused by the most recent operation. The stack register also known as the T-register is a 32-bit register that contains a pointer to the top of a pushdown stack associated with the currently active procedure. Stacks to be described infra provide a work space, and a mechanism for saving local variables and preserving procedure entry, and return information. The boundary address register (BAR) 206 is a 28-bit register which specifies the lowest absolute main memory address accessible by software. The register is loaded during system initialization and can only be read by software. The hardware control mask register 208 is an 8-bit register which records machine condition information.

J. INSTRUCTION FORMATS

There are approximately 200 instructions although more or less may be utilized. Each instruction is one of

four different lengths but always an even number of bytes long. Instructions are stored in consecutive storage locations. The address of the leftmost byte is a multiple of 2, and is the address of the instruction.

The eight most significant bits (and in some cases bits 8 through 11 or 12 through 15) of an instruction represent the operation code, while the remaining bits represent one or more operands. An operand may be a register designator, displacement designator, address syllable (logical address) literal value, immediate literal value. The type and number of operands are determined by the instruction format.

III. SYSTEM ORGANIZATION

A. JOB STEP AND TASK

Work to be performed by the computer system is defined externally by a series of job steps via a job control language. A job step is a unit of work to which hardware resources are allocated. Typically a job step consists of several tasks. A task is the smallest unit of user defined work consisting of a stream of instructions executed without parallelism.

B. PROCESS

The user-visible concepts of task and job step are represented in the hardware by a process and process group, respectively. A process is defined as an ordered sequence of instructions which can be executed asynchronously by the CPU (i.e., several processes can be active and sharing resources, but only one process is actually running at any one instant). A process group is a related set of processes necessary to perform one job step.

C. PROCESS CONTROL BLOCK AND SYSTEM BASE

Because processes can relinquish CPU control at various points during their execution, a storage area in main memory is made available to a process to save CPU

status. This status information is utilized to precondition the CPU before a process regains control of the CPU.

The storage area assigned to a process is called a process control block (PCB) 400 on Figure 4. The data contained in a PCB include the address of memory areas (address space) assigned to the process, the contents of all pertinent registers, and the state of the process. Thus a PCB serves as a temporary storage area for information necessary to start or restart a process without any information loss. Each PCB is visible to the hardware and can be addressed by the operating system via a set of hardware tables developed during system initialization and modified during system operation (Figure 5).

There is an absolute main memory area which is referred to as the system base (Figures 5 and 6). This area is developed by firmware and is accessible via the base address register (BAR) 501 which can be read but not written. The system base 502 contains a number of system attributes which include a job step number and a process group number (J, P) respectively for the currently running process. Another attribute in the system base is a pointer to a hardware defined data structure known as the J table 503. This table contains an entry for every job step presently in the system. Each entry in the J table 503 points to an associated P table 504 which is also a hardware defined data structure. This table defines a process group and contains an entry for every process in the process group. Each P-table entry points to a PCB 400.

Referring to Figure 5 the J-table pointer 505 indexed by the J number via computational unit 506 provides access to a J-table entry 503. This entry contains a P-table pointer which when indexed by the P number via computational unit 506 provides access to a P-table entry 504. The P-table entry contains a pointer 507 to the PCB of the current running process. Thus the operating system can access the active PCB using the contents of the BAR 501 and can access any other PCB given its associated (J, P) logic name.

D. MEMORY SEGMENTATION

In a multiprocess environment, such as herein described there are many processes in memory at any given time. These processes vary in size and demand for memory which causes a memory allocated problem. The hardware herein described in cooperation with an operating system (not shown herein) solves the problem by dynamically allocating memory space. Due to the random nature of memory requirements, memory is allocated in variable size segments and the memory allocation can be restructured during process run time. Thus, a process may be allocated a number of noncontiguous memory segments. This memory allocation method is called segmentation.

Segmentation presents an additional problem in that memory addresses have to be modified whenever part or all of a process is relocated. To alleviate this problem the system herein described provides a technique whereby addresses used by a process are logical rather than absolute main memory addresses. These logical addresses are used to develop absolute addresses.

Segmentation also allows each process to access its own or related memory segments via a system of segment descriptors. By accessing a segment descriptor, a process can obtain the address of a segment. Segment descriptors are contained in main memory and are maintained by the operating system.

Each process may have access up to 2068 memory segments. Normally, this would require an equal number of segment descriptors per process. However, since segments can be shared, the operating system groups segment descriptors into segment tables. This grouping is based on accessability by one process (task), a process group (job step), or globally (system wide). Each process may have up to 15 segment tables associated with it. This technique requires only one segment descriptor for each segment which can be accessed by a process via a segment table. Thus, the memory space required for segment descriptors is decreased; memory updating during relocation is reduced; and some program protection

is provided. (The main mechanism for program protection is the ring system. See Patent Application No. ———— filed ———— entitled "Protection of Information in a Multiprogram, Multiprocessor Computer System;" also see Patent Application No. ———— filed ———— entitled "Ring Crossing Mechanism" both assigned to the same assignee as the instant invention.

A process must be able to determine which segments it is allowed to access. Accordingly, the system provides a process with two segment table word arrays (STWA). These arrays contain the addresses of all segment tables accessible to a process. There are two segment table word arrays per process because there are two segment sizes, large and small. Large segments have a maximum size of 2^{22} bytes while small segments have a maximum size of 2^{16} bytes. All segments vary in size in 16-byte increments up to the maximum. A system can typically accommodate up to 28 large segments and 2040 small segments.

Segment table word arrays may be relocated by the operating system; therefore, a process must know the absolute address of its associated STWA's. The PCB for any process contains two words which contain this information which are known as address space words ASW0-1 on Figure 4. Each word points to a segment table word array STWA. The operating system updates the contents of the ASW's whenever the associated STWA's are relocated. Working down the chain of pointers and decoding the segment descriptor is a firmware function and thus once initiated is not visible even to the operating system.

Segmentation defines over 200 million bytes of address space as being available for processes. This number exceeds the capacity of main memory; therefore, a secondary storage (magnetic disk or drum) is used in conjunction with main memory. The operating system creates the illusion that the system has a much larger main memory than is really available. This concept is called virtual memory.

At any given time, a defined segment may or may not be physically in main memory. The contents of a seg-

ment descriptor indicates whether or not the associated segment is in main memory. The hardware detects any attempts by a process to access a segment not in main memory and notifies the operating system. The operating system causes the desired segment to be loaded into main memory from secondary storage. Then the operating system places the segment's memory address in the segment descriptor which is the only place where the absolute address of a segment can be found. This operation is invisible to the process and thus it is not aware that the segment was not in main memory or that it may have to be relocated in main memory. (For details on memory segmentation see Patent Application No. ———— filed ———— and entitled "Segmented Address Development". Also see patent No. ————.)

The computer system herein described provides data and procedure protection by preventing processes from interfering with each other or sharing each other's address space in an unauthorized manner. This protection is accomplished by restricting addressability via memory segmentation and by a ring system.

The segment tables isolate the address space of the various processes in the system. Processes always use a segmented address during execution. A segmented address consists of a segment number and a relative address within the segment (see above referenced application on Segmented Address Development). The hardware checks that the address used by a process is part of the address space assigned to the process. If the address is outside the prescribed address space, an exception occurs. A process cannot refer to data within the address space of another process because the hardware uses the segment tables of the referencing process. Thus, there is no possibility for a process or process group to reference an entity belonging to another process group.

Generally, overlap in address space in the system occurs for those segments shared by all processes. These public segments are created by system programs which check to insure against address conflicts. Thus, segmentation protects user programs against each other and protects the operating system against user programs.

Segments shared by several processes are not protected from misuse by one of these processes. To solve this problem, a ring system is utilized whereby procedure and data segments are grouped into a four-class hierarchy. The four ring classes are numbered 0 through 3. Each ring represents a level of system privilege with level 0 (the innermost ring) having the most privilege and level 3 (the outermost ring) the least. Every procedure in the system has a minimum and a maximum execute ring number assigned to it which specifies who may call the procedure. A procedure is a subroutine which is capable of calling other procedures and passing parameters to them:

The general rules of the ring system are as follows:

1. A procedure in an inner ring has free access to data in an outer ring. Conversely a procedure in an outer ring cannot access data in an inner ring.
2. A procedure in an outer ring can branch to a procedure in an inner ring, but the reverse is not allowed.
3. Each segment containing data is assigned two ring values, one for read (RD) and one for write (WR). These ring values specify the maximum ring value in which a procedure may execute when accessing the data in either the read or write mode.

Each time a procedure instruction is executed, the procedure's ring number (effective address ring, EAR) is checked against the ring numbers assigned to the segment containing the referenced data. The EAR is the maximum number of process ring numbers in the instruction counter and all ring numbers in base registers and data descriptors found in the addressing path. Access to the data will be granted or denied based on a comparison of the ring numbers. For example, if a system table exists in a segment having a maximum read ring value of 3 and a maximum write ring of 1, then a user procedure executing in ring 3 may read the table but may not update the table.

By predesign, rings 0 and 1 are reserved for the operating system and rings 2 and 3 are reserved for the

user. Ring 0 contains those segments critical to total system operation. Ring 1 contains the bulk of the system segments whose failure would not be catastrophic and would allow recovery. The user may utilize ring 2 for checked-out programs and ring 3 for programs being debugged.

F. PROCEDURE CALLS

The procedure call is an important function in the system herein described. Procedure calls are used to pass from one procedure to another, to allow user procedures to employ operating system services; and to achieve a modular structure within the operating system. A procedure call is effected by instructions and a hardware recognized entity called a stack (Figure 7A).

A stack is a mechanism that accepts, stores and allows retrieval of data on a last-in-first-out basis. Stacks reside in special segments called stack segments. A stack segment consists of a number of contiguous parts called stack frames 701 (Figures 7A and 7B) which are dynamically allocated to each procedure. The first stack frame is loaded into the top of the segment and succeeding frames are loaded after it. The last frame loaded is considered the top of the stack. The T-register 702 locates the top of the stack for the currently active process. A virtual T-register exists on the PCB of all other processes in the system.

A stack frame 701 of Figure 7B consists of three areas: a work area 702 in which to store variables, a save area 703 in which to save contents of registers, and a communications area 704 in which to pass parameters between procedures. Prior to a procedure call, the user must specify those registers he wishes saved and he must load into the communications area the parameters to be passed to the called procedure. When the call is made, the hardware saves the contents of the instruction counter IC and specified base registers to facilitate a return from the called procedure.

Each procedure call creates a stack frame within a stack segment 701 and subsequent nested calls create additional frames. Each exit from one of these called

procedures causes a stack frame to be deleted from the stack. Thus, a history of calls is maintained which facilitates orderly returns.

To insure protection between procedures executing in different rings, different stack segments are used. There is one stack segment corresponding to each protection ring per process. A PCB contains three stack base words which point to the start of the stack segments for rings 0, 1 and 2 associated with the process. The ring 3 stack segment can never be entered by an inward call; therefore its stack starting address is not required in the PCB.

IV. PROCESS MANAGEMENT AND SYNCHRONIZATION

The system herein provides for multiprocessing operations which are controlled by an operating system using a combination of software, hardware and firmware. Software creates and deletes processes within the system while hardware and firmware multiplex processes on the CPU. In addition, a combination of software, hardware and firmware provide for synchronization between processes.

Processes are normally, but not always, started and stopped at the initiation and termination of i/o operations, during related job handling, and at other times for purposes deemed necessary by the operating system. Therefore, a comunications system is necessary to efficiently start and stop related processes and to pass information between them. The hardware system herein provides internal messages called semaphores to provide a communications link between the processes.

A. PROCESS STATES

A process can be in one of four possible states at any time: running, ready, waiting or suspended. The hardware recognizes these four possible process states and executes various firmware procedures to effect process dispatching, state changes and to maintain data structures based on a process's state. The PCB contains a

state field which defines the current state of its associated process.

A process is in the running state when it has control of the CPU. This state involves supplying the CPU with an address space (segment tables) and a starting address. The CPU then executes instructions in the procedure segments of the process. The process name J table word (logical address) of the PCB for the currently running process is retained in the running process word (BAR +60) within the system base (Figure 6). (Note: The system base shown in Fig. 5 is the same as that shown in Fig. 6, but with some details omitted.)

The ready state is equivalent to running state except that the process does not have control of the CPU because it has not been recognized by the CPU. A process in the ready state is in contention for the CPU with other ready processes and the running process.

A process is in the wait state when it cannot continue until a specific event occurs such as a message via a semaphore. A waiting process is not in contention for the CPU but it may be in contention with other waiting processes for the required event.

A suspended process is a process which has been stopped for a time by software and may be resumed later. The decision to stop and resume the process is external to the process. Thus, a suspended process is not active and therefore cannot receive notification of event occurrences and cannot utilize the CPU.

A process is suspended under the following conditions:

(1) by executing a Terminate instruction (as a result of having completed all its functions.)

(2) By execution of a Suspend instruction by the operating system.

(3) By the occurrence of an exception condition whereby control is transferred to the operating system.

B. PROCESS DISPATCHING

Processes move from one state to another voluntarily by action of the process while running or involuntarily by the actions of other processes. CPU firmware, known

as the dispatcher, controls the transaction of processes between states. The dispatcher uses a set of queues (to be later described) to manipulate processes which are in the ready or the waiting states. Suspended processes are controlled by software.

Referring to Figures 6, 8 and 9, a ready or waiting process is represented by a PCB and a special queue entry called a process link. Figure 9 shows an exploded view of contents of the GO segment 802, and contains process links 803a-803b and 803c-803g of active processes, and free process links 805a-805c of suspended processes. Each process link specifies the process name (J, P), the process priority and a pointer to the next process link in the queue. There are various types of queues such as wait queue 803a-b and ready queue 803c-g.

A hardware device similar to the J table, known as the G table, (Figures 6 and 8) contains pointers to all general (known system wide) segments 802-802n. The first element, GO, of the G table 801 points to that segment 802 containing the dispatcher queues. A G-table pointer to the G table 801 is found in the system base 502 on Figure 5. Also in the system base is an entry called the internal process queue word (IPQW) which identifies the head 805 of the ready queue 803c-803g in the GO segment 802.

Thus, the dispatcher can examine all ready processes by consulting the ready queue 803c-803g. When the currently running process changes states, the dispatcher removes the process link at the head of the ready queue and uses the J, P name to access its PCB. The process defined by the PCB then becomes the new running process.

Since more than one process may be awaiting on the same event, a queue of waiting processes 803a-803b exists for each event. Waiting processes are also strung together via process links 805 residing in the GO segment. A pointer to the head of a wait queue exists in a semaphore 903 (to be later described). A number of events exist for which a process may wait; therefore, there are a number of wait queues each of which has an associated semaphore 903, 904.

The number of processes ready or waiting varies dynamically. Thus, the number of process links required for the ready and wait queues also varies. This fact introduces a mmeory management problem for the dispatcher The problem is solved by another queue called the free process link queue 805a-c. This queue links together all process links is segment GO that are not being used by the ready or the wait queues and can be used to extend a particular queue of ready or waiting processes. A pointer 901 to the head 902 of the free process link queue 805 resides near the beginning of the GO segment 802.

C. PROCESS SYNCHRONIZATION

Process synchronization is required to coordinate the activities of two processes working on the same task. The synchronization is achieved using semaphores 903-904 which are data structures residing in the address space of communicating processes. A semaphore is used to signal event occurrence and to handle queues of messages. An event in this context is anything observed by a process which may be of interest to some other process. The event may be the completion of an asynchronous operation or the availability of a resource.

A process uses two semaphore operations to signal an event occurrence. One operation sends a signal to a semaphore; the other picks up a signal from a semaphore. (The sending operation is often called a V-operation; the receiving operation is called a P-operation). The sending operation allows a process to send data or a signal that data are ready. The semaphore stores the signal until another process is ready to pick it up. Thus, the sending process is free to proceed, since it has sent the data. The receiving operation examines a specified semaphore and picks up the signal. If a signal is present, the receiving process continues executing. However, if there is no signal at the semaphore, the receiving process enters the wait state. The semaphore then serves as a pointer to the head of a wait queue. The process remains in the wait state queued at the semaphore until another process sends a signal to that particular semaphore.

Thus, a semaphore can hold a signal until a process picks it up, or a semaphore can hold a process until a signal is sent to it.

Messages can also be passed from process to process. A message has the same present or not present quality as a signal plus additional information. Part of the information is supplied by hardware and part is supplied by the procedure of the process that sent the message. A message carries the process name of the sending process. Thus, many processes can send information through a single semaphore stamped with the sender's name.

A message semaphore may have a queue of messages waiting to be picked up by processes. As with signal semaphores, requirements for memory space increases and decreases thus presenting a memory management problem. Again, the problem is solved with a queue of free message links. These links reside in a known place in a segment that can easily be found when needed to supply or absorb message links.

Because semaphores and the queues built on them are shared by different processes, the total semaphore structure is protected. This is accomplished by hardware and software conventions that restrict access to any segment containing semaphores. Thus, semaphores must be in semaphore descriptor segments, some of which may be G segments (if system communications is necessary). However, all G segments (except GO) are semaphore descriptor segments.

Each semaphore descriptor contains a pointer to a semaphore. Semaphore addresses are developed via a semaphore descriptor, thus providing added protection for the semaphore. A semaphore segment can be addressed logically using a segment number and a relative location within the segment or directly using the G, D number.

PROCESS CONTROL BLOCK STRUCTURES

Referring to Figure 4 there is shown the format of the process control block (PCB). The process control block 400 is a storage area in main memory made available to a

process to save the CPU status. Addressing a PCB is performed as described supra in relation with Figure 5. The PCB pointer 507 (Figure 5) points to the process control block PCB at memory location 0 on Figure 4. It will be noted that proceeding in a downward direction memory locations increase by 4 bytes whereas in proceeding in an upward direction from memory location 0 they increase by 8 bytes. The downward memory locations are considered positive from 0 whereas the locations in an upward direction from 0 are considered negative directions. The upward locations are optional and may or may not be included in the process control block; also locations 148 through 176 are also optional. (Note that the numerals under memory location specify the displacement in bytes from the 0 reference location of the process control block PCB and are not to be confused with the reference numerals commonly used to identify parts in a patent drawing). Starting at byte 0 up to but not including byte 16 there are stored four process main words PMW 0 through PMW 3 with each process main word PMW being four bytes in length. Process main word 0 occupies bytes 0 through 3 and is comprised of 4 parts: a capability byte, a priority byte, a state byte and a decor extension byte DEXT. Referring to Figures 10a through 10d there are shown details of process main word PMW 0, with further details of the capability byte 1001 shown on Figure 10b. Referring to Figure 10b, the first bit 1005 is the accounting mode bit for indicating whether or not time accounting functions are being performed for the process. When the accounting mode bit 1005 is set to binary 0 no time accounting function is being performed for the process; whereas when the accounting mode 1005 is set to binary 1, time accounting is being performed. The scientific mode bit 1006, when set to zero, indicates that saving of scientific register of the machine is not performed and the scientific register saving area located at bytes 148 to 176 on Figure 4 does not exist in the process control block PCB. When the scientific mode bit 1006, is set to binary 1, the scientific optional feature exists and is being used in the process, and the scientific registers savings area is used to save the contents of the scientific registers when necessary.

The code mode bit 1007 indicates whether or not a standard code set or compatibility code set is being used by the process, with a binary 0 in that position indicating that standard code set is being used; whereas a binary 1 in the third bit position 1007 indicates a compatibility code set is being used. The remaining of the bits of the capability byte are set to zero.

Details of the priority byte 1002 are shown on Figure 10c. Referring to Figure 10c the first four bits 1008 of priority byte 1002 is utilized to set the priority level of the process associated with that given process control block PCB. Each process is assigned one of 16 levels of priority which is used for ordering competing processes i.e. (a) for choosing the process to be run among ready processes, (b) for putting processes in queues. Priorities decrease from 0 to 15, and for a given priority level the FIFO (first in first out) rule is applied. The next 4 bits 1009 of priority byte 1002 are zeroes.

Referring to Figure 10d details of the state byte 1003 are shown. A state byte is utilized to provide information with regard to the process associated with the process control block PCB 400. The active field bit A 1010 is set to binary 1 when the process is activated. The suspend field S 1011 is set to binary 1 when the process is suspended. The substate field SS 1012 is a 2 bit field and defines the following substates of the process: (a) when set to binary 00 the process is inactive; (b) when set to binary 01 the process is waiting in the queue of ready process (Q/PR/RDY); (c) when set to binary 10 the process is waiting on a semaphore in a queue of semaphores (Q/PR/S); (d) when set to binary 11 the process is being executed by the processor. The mid-operation field (MOI) 1013 is set to binary 1 when an interrupt happens and is taken care of during the execution of an instruction—i.e. before the completion of the process. The extended decor mode bit EXTD 1014 is set to 1 when the process is operated in an extended decor mode which is an emulation mode of the machine. Bits 1015 and 1016 are set to 0. The fourth byte of process main word PMW 0 contains the decor extension number and is utilized when the system is in emulation mode.

Process main word PMW 1 is stored in bytes 4-7 of the process control block PCB. Details of PMW 1 is shown on Figure 10e. The status byte 1016 is the first byte in PMW 1 and stores the status register contents. The multiprocessor byte MP 1018 is significant in a multiprocessor architecture otherwise this field is zero. The second and fourth bytes of process main word 1 are the MBZ fields 1017 and 1019 respectively which must be zero for normal operation.

Process main word PMW 2 occupies bytes 8 through 11 of the process control block and is shown in more detail on Figure 10f. Referring to Figure 10f the field from bit 4 through bit 31 contains the local name SEG, SRA 1021 of the semaphore to which the PCB is linked when the process is either in the waiting or suspended states. The exception class and type field 1023 contains the class and the type of the interrupt-like exception which cause the process to enter the suspended state after an exception. The field from bits 4 through 15 is meaningless 1022 when a process is in a different state than those mentioned above.

Process main word PMW 3 occupies bytes 12 through 15 in PCB 400 and points to a decor extension table. Referring to Figure 10g for details of PMW 3 the DETSZ field 1024 defines the number of entries in the table and if this field is zero no decor extension is allowed to the process. The DETA field 1025 is the absolute address of the decor extension table in units of 16 byes and is significant only if DETSZ is not 0. The decor extension table is made up of DETSZ entries. Each entry is one byte size. The DEXTth entry of the table defines the capability of the process to operate in the decor extension mode DEXT. When the DEXTth byte is 0 the decor extension number DEXT is not allowed, whereas if the DEXTth byte is 1 the decor extension number DEXT is allowed. Values of DEXT other than 0 and 1 are illegal. (See Figures 10a DEXT number 1004).

Bytes 16 through 23 of PCB 400 contains 2 address space words ASW 0 and ASW 1 respectively and each ASW contains a pointer to an array of segment table words. Both ASW 0 and ASW 1 respectively have the

same format shown on Figure 10h. The size of the array of the segment table words is defined by the number of segment table words in an array and typically comprises six for ASWO and eight for ASW 1. The STWSZ field 1026 indicates the size of the array of the segment table words. The segment table word array field STWA 1027 contains the absolute address STWA of the array in units of 16 bytes—i.e. the absolute address of the array is 16 times STWA in bytes.

Bytes 24 through 27 in the PCB contain an exception word EXW shown in greater detail on Figure 10i. The exception word contains a pointer (SEG, SRA) 1029 to an exception class table which defines the action to be taken following a process exception according to its class as stored in process main word PMW 2. (See Figure 10f). The MBZ field 1028 of exception word EXW must be 0.

The stack word SKW located in bytes 28 through 31 of the PCB contains the value of the top of the T register of the stack of the process when the process is not running and is shown in greater detail in Figure 10j. Referring to Figure 10j, bits 0 and 1 define the TAG field 1030. The TAG indicates the type of descriptor by its contents and must be zero for SKW. Bits 2 and 3 of the SKW word contain the RING field 1031 which contains the ring number associated with the segmented address of the stack for protection purposes and in this case must be zero. Bits 4 through 31 contain the segment number SEG, and the segment relative address SRA 1032 and is a field which identifies the segment described in a segment table and the segment relative address within the segment. The stack word SKW is updated every time the process leaves the running state. It is used to restore the T register contents every time the process becomes running. In this last case the TAG 1030 and RING 1031 are tested to be zero, otherwise an illegal PCB exception occurs.

Bytes 32 through 35 of the PCB 400 contain the instruction counter content word ICW sometimes also referred to as ICC. Referring to Figure 10k there are shown details of the instruction counter word ICW wherein the TAG field 1033 must contain binary 00 (i.e.

values other than zero are illegal in the instruction counter). The current RING field 1034 occupying bits 2 and 3 defines the current ring number of the process to be used in determination of access rights to main storage. Bits 4 through 31 define the segment number and the segment relative address (SEG, SRA) 1035 which define the address of the next instruction to be executed.

The MBZ field in bytes 36 through 39 must be zero. (Note the MBZ field always indicates a field which must be zero). The MBZ word is tested every time the PCB is accessed from the name J, P. If it is not zero an illegal PCB exception occurs.

Stack base words SBW 0-2 occupy bytes 40-51 in the process control block 400. These words have the same format which is shown in greater detail on Figure 101. They are utilized during stack operations and whenever used their TAG field 1036 and RING field 1037 must be zero otherwise an illegal PCB exception occurs. Bits 4 through 31 contain the segmented address (SEG, SRA) 1038 of the first bytes of the stack segments for ring zero, 1 and 2 respectively.

Bytes 52 through 83 of the process control block 400 is a space reserved for the base registers saving area (8 words). Bytes 84 through 147 is a saving area which is utilized to save the values of all general registers (16 words). Bytes 148 through 179 is a saving area which is utilized to save the scientific registers (8 words).

Five double words are provided in the PCB 400 above the PCB zero address, for time accounting purposes when the accounting mode bit in the PMW 0 word is set. These words are located from PCB address minus 8 to PCB address minus 40. Each word contains a time or a time interval expressed in microsecond units in its first 52 bits with bits 52-63 filled with zeroes. The residual time out double word RTO (first 8 bytes above 0 in the PCB) contains the quantum of time which is actually spent by the processor on behalf of the process before a time out exception occurs. The RTO word is updated in the following way: each time the process exits the running state the process timer value is stored in the RTO word. Each time the process enters the running state, the process timer value is loaded from the RTO.

The running time accounting RUA double word at bytes 7 through 15 is a time counter which specifies the total amount of processor time a process was in the running state. The time accounted for is the time actually spent by the processor on behalf of the process exclusively. The RUA word is updated in the following way: each time the process exits the running state, the value of the process timer PT is read. The difference of the contents of RTO and PT is added to RUA. (Consecutively, the PT value is stored in RTO). Note that the time during which the process is suspended is not computed. The RTO and RUA words are updated even if the accounting mode bit is set to 0. However the CET, RTA, and WTA words (to be later described) are provided in the process control block only if the accounting mode bit in the process main word PMW 0 is set to 1. They are updated only in this case.

The waiting time accounting WTA word at bytes 17 through 23 is a real time counter which specifies the total amount of real time the process was in the waiting state. The WTA word is updated in the following way: each time the process exits the waiting state the time of day clock (not shown) value TOD is read and the Value of TOD minus the value of CET word is added to the WTA word.

The ready time accounting RTA word located at bytes 24 through 31 is a double word which is a real time counter which specifies the total amount of real time the process was in the ready state. The RTA is updated in the following way: each time the process exits the ready state, the time of day clock value TOD is read, and the contents of TOD minus the contents of CET is added to RTA.

The current entry time CET double word at bytes 32 through 39 contains the time of day at which the process entered one of the following states: ready, waiting, running, and suspended.

SYSTEM BASE STRUCTURES:

Referring to Figure 6 the format of the system base 600 is shown. The system base resides in absolute main memory and is developed by firmware and is accessible

via the boundary address register (BAR) which can be read but not written. The boundary address register BAR is below an area in main memory reserved for hardware and separates this area in memory reserved for hardware and the system base 600. Referring now to Figure 6 the system base 600 contains a number of system attributes which includes a job step number and a process group number (J, P) for the currently running process. From the logical name of the process J, P, the absolute address of the corresponding process control block PCB is obtained. The size and address of the J table are defined by the contents of the J table word (JTW). This word is located at the address defined by the BAR register. The format of the JTW is shown on Figure 11a. The size (JTSZ) 1101 or the J table 1204 on Figure 12 defines the number of entries in the J table 1204 which may be up to 255 entries. The JTSZ 1101 is an 8 bit positive integer; an out of J table exception occurs if J is greater than JTSZ. The absolute address of the J table 1204 is obtained by multiplying the J table pointer 1102 by 16. The J table 1204 contains J table entries whose format is shown in greater detail on Figure 11b. Each J table entry defines the absolute address of a P table 1205 which is obtained by multiplying the P table pointer 1104 by 16. The size (PTSZ) 1103 of a P table defines the number of entries in the P table. The PTSZ is an 8 bit positive integer which may typically vary from 0 to 255 to indicate the number of entries in the P table. An out of P table exception occurs if P is greater than PTSZ. Each entry of the P table 1205 defines the absolute address of a process control block (PCB) 1206 by multiplying the process control block pointer 1107 by 16. A presence indicator P 1105 indicates the absence of a PCB 1206 when set to binary 0 and indicates the presence of a PCB when set to binary 1. (When the presence indicator P 1105 is found to be 0 a vacant P table entry exception occurs). Bits 1 through 7 of the P table indicator (Figure 11c) must be 0 (MBZ) 1106, otherwise an illegal P table entry exception occurs.

At address BAR plus 4 of the system base 600 there is the format byte of a G table word (GTW) shown in

greater detail on Figure 11d. The size and the address of a G segment-table 1212 on Figure 1200 are defined by the contents of the G table word (GTW). The size (GTSZ) 1108 of the G table 1212 defines the number of entries in the G table which may typically be up to 255 entries. GTSZ is an 8 bit positive integer; an out of G table exception occurs if the G number is greater than the GTSZ. The absolute address of the G table 1212 is obtained by multiplying the G table pointer 1109 by 16. The format of the G segment table entry has a two word size (8 bytes) and is called a G segment descriptor. The format of the G segment descriptor is shown in detail on Figures 11e and 11f. All G segment descriptors are direct and therefore the indirect bit I, 1111 must be 0 otherwise an illegal segment descriptor exception occurs. The presence indicator P 1110 is a one bit field which when set to binary 1 indicates that a segment is defined in main storage for the segment number to which that descriptor corresponds; whereas if it cleared to 0 no segment is defined and a reference to the segment descriptor causes a missing segment exception. The available bit A 1112 is a one bit field which indicates whether or not the segment is available; it is only checked if this segment is defined (i.e. P equals binary 1), otherwise it is ignored. The used flag field U 1113 indicates whether or not the segment has been accessed. If the U bit is set to binary 0 the segment has not been accessed; whereas if the U field is set to binary 1 the segment has been accessed. The written flag field W 1114 indicates whether the segment has been written. If W is set to binary 0 the segment has not been written; whereas if W is set to binary 1 the segment has been written. The gating indicator GS 1115 of a G segment descriptor must be set to binary 01, otherwise an illegal segment descriptor exception occurs. The reason for this is that a G segment always contains semaphores (although the reverse is not true i.e. all semaphores are not required to be in a G segment) and instructions on semaphores require the GS code to be binary 01. The absolute address of the base of a segment 1214 is defined in the G segment descriptor of Figure 11e by the 24 bit base field 1116; the content

of this field is multiplied by 16 to obtain the absolute address. The second word of the G segment descriptor of Figure 11f occupies bit position 32 through 63 in the G table 1212. The RSU field 1117, bits 32 through 39 is reserved for software use and is generally ignored when used as a G segment descriptor as it is in this case. The MBZ field 1118 must be 0 otherwise an illegal segment exception occurs. Since the MBZ field 1118 occupies bits 40 through 51 it sets the SIZE field 1119 which is the field for a small segment SIZN; hence all G segments must be of the small segment type. The segment SIZN 1119 is a 12 bit positive integer defining the number of bytes in the segment and the segment size is interpreted as a multiple of 16. Therefore the segment size for a G segment 1214 cannot exceed 2^{16} bytes (small segments).

Referring once again to the system base 600 of Figure 6 there are 9 system exception cell words located between BAR plus 8 and BAR plus 44. The format of the system exception cell words EXC is shown on Figure 11g. Since semaphores are utilized for transmitting messages to dedicated processes when a system exception occurs the pointers to these semaphores are found in 9 locations of memory each location called a system exception cell—one per class of system exception. The MBZ field 1120 must be set to binary 0 otherwise a system check occurs. Each exception cell (EXC) contains the system name G, D 1121 and 1122 respectively.

The channel exception cell located in BAR plus 44 of the system base 600 has a format which is similar to the system exception cell previously discussed and contains the system name GD of a semaphore which is used for transmitting messages to dedicated processes when a channel exception occurs.

An internal processor queue word IPQW is located beginning at BAR plus 48 and details of its format are shown on Figure 11h. The IPQW word points to the head of a queue of processes ready (Q/PR/RDY) as shown on Figure 9 by reference numerals 905 and 805. The queue of processes ready (Q/PR/RDY) links all processes which are in the ready state. It is referenced by the HEAD

OF Q/PR/RDY-field 1124 (Figure 11h) of the IPQW word by pointing to the top of the ready process queue. The HEAD of Q/PR/RDY-field 1124 contains a 16 bit positive integer which is the displacement from the base of the G segment number 0, referred to as the GO segment, to the first byte of Q/PR/RDY. If this Q/PR/RDY bit field is 0, the ready queue is considered to be emtpy. The MBZ field 1123 must be 0 otherwise a system check occurs.

At BAR plus 52 of the system base 600 there is shown the storage for the initial and current retry counts whose format is shown in detail on Figure 11i. The NFS field 1125 is a nonfunctional storage field and is not utilized by the system base. The initial retry count field 1126 and the current retry count field 1127 are used to control the number of times automatic instruction retry is executed before a machine error is made to produce a machine failure exception condition. They are loaded with the same number by a Reset Retry Count (not shown herein).

The running process word (RPW), shown in Figure 11j, is stored in BAR plus 56 of the system base 600 and is used to store the name of the running process with its priority in case of a monoprocessor architecture. The NFS fields 1128 and 1131 respectively are nonfunctional storage fields and may be utilized for any purpose by any facility but is generally not utilized by the system base. The priority level of a running process is stored in the PRI field 1129. An asynchronous trap bit is stored in AB field 1130; whereas an asynchronous trap ring is stored in ARN field 1132. The logical name J, P of the running process in case of a monoprocessor architecture is stored in the J, P field 1133.

An Absolutization Table Pointer word shown on Figure 11k is located at BAR plus 60 in the system base 600 and is utilized in initial system load to initialize the absolute addresses in the initial system load (ISL) program by adding the contents of BAR to all absolute addresses in the ISL program. The Absolutization Table Pointer 1135 defines the location of an Absolutization Table (not shown). The Absolutization Table Size is shown by the ATSZ field 1134.

The CPU serial number word shown on Figure 11l is a 4 byte word located at BAR plus 64 and contains the serial number of the CPU in the CPU serial number field 1136.

A main storage upper limit word shown on Figure 11m is located at BAR plus 68 and indicates the main storage upper limit 1139 by providing the absolute address of the last available word in main storage.

At BAR plus 72 there is located a word shown on Figure 11n for providing the initial system load ISL device channel number (CN) 1140 and the hardware device channel number (CN) 1141.

The type and subtype of a device used in the computer system is shown by a hardware device type word (Figure 11o) in fields 1143 and 1144 respectively; where the RSU field 1142 is reserved for software. This word is found in the system base at BAR plus 76. A similar word having a similar type format shown on Figure 11p contains the type and subtype of the device used in the initial system load. This word is located at BAR plus 80.

When the restart button of a computer is pressed, a simulated V-operation is performed on a semaphore and the Ready state is entered. A pointer to this semaphore is found at BAR plus 84 of the system base 600 and is called a restart cell word, and has a format shown on Figure 11q. The format is similar to the system exception cell described supra and contains the system name G, D of a semaphore in the G field 1149 and D field 1150 respectively. The MBZ field 1148 must be 0.

When there is more than one processor to the computer system, a word is provided in the system base 600 at BAR plus 88 for multiprocess extension. Details of this word are shown on Figure 11r.

Examples of System Base and Process Control Block Use

Referring to Figure 12 there is shown one example, how the system base may be utilized in combination with the process control block in order to address and access a user segment, a system segment, or a queue of processes read (Q/PR/RDY) segment. Main memory 1200 has a

portion 1203 reserved for hardware use. A boundary address register BAR 1202 separates the system base 1215 from the portion of memory 1203 reserved for hardware. The boundary address register BAR 1202 is utilized to address items in the system base 1215 by adding the contents of the boundary address register to the displacement in 4 byte units of the item desired in the system base. This address then points to the first byte of the item in the system base desired. In Figure 12 the BAR 1202 is pointing at the J table word (JTW). The J table word, as previously discussed, has a pointer which points to a J table 1204. By indexing to the J number shown on Figure 5, a J table entry 1216 is obtained. At the J table entry there is a P table pointer which points to the absolute address of the P table 1205. By indexing to the P number (see Figure 5) within P table 1205 the absolute address of the process control block 1206 is obtained. As previously shown in process control block PCB 1206 there are two address space words ASW 0 and ASW 1. The high order bits of the segment table number field STN in the base register 1201 is used to access one of these two address space words, in this instance ASW 1 which has a segment table word array STWA pointer that points to segment table word array STWA 1208. Together with the segment table number STN of the base register 1201 one of 8 segment table words is accessed in STWA 1208, which points to one of 8 segment tables 1210. The segment table entry STE from base register 1201 is then utilized to make one of 256 entries in segment table 1210 where a segment descriptor is located. The segment descriptor is then utilized to access a user segment 1211. (For greater detail see Application No. ———— filed on ———— (entitled Segmented Address Development and assigned to the same assignee as the instant invention and is hereby incorporated by reference herein).

In order to access a system segment 1214 which is utilized to store semaphores a G table word GTW is utilized in the system base 1215. The address of the G table word is obtained by adding the displacement of the G table word in the system base to the boundary

address register BAR 1202. (See Figure 6). The G table word GTW includes a G table pointer which points to a G table 1212. By utilizing a G number available to the system and indexing in the G table a G segment descriptor is accessed which is utilized to address a system segment 1214.

Similarly the system base 1215 is utilized to access the queue of process ready (Q/PR/RDY) 1213 by locating an internal processor queue word IPQW which points to the Q/PR/RDY segment 1213.

Control Unit

Referring to Figures 13a-13c details of the control unit are shown. The control unit, although shown separate from the central processing unit (CPU), is in actuality a part of the CPU and is comprised of a control store unit CSU 1301, a control store interface adapter CIA 1302 and appurtenant subunits, control store loader CSL 1303 and control and load unit CLU 1304.

The control store unit CSU 1301 receives microinstructions from the control store loader CSL 1303 via the control and load unit CLU 1304 and the control store interface adapter CIA 1302. Under normal operating conditions, microprograms are loaded from an external source during system initialization and become a permanent control function of the machine. However the control store unit CSU 1301 has the ability to be reloaded and initialized in a manner that provides for a variety of central processing unit CPU 1306 operational modes. The following modes of operation of the CPU are available under control of the CSU 1301; (a) native mode; (b) emulation mode; (c) concurrent native and emulation modes; (d) diagnostic mode. This capability is possible because the micro-instructions resident in the CSU are the source of micro-operations used to control the operation of all other CPU functional units such as the emulation unit 1316, the arithmetic logic unit ALU 1317, the instruction fetch unit IFU 1318, the address control unit ACU 1319 and the data management unit DMU 1321. Also shown within the central processing unit CPU 1306 are previously described general regis-

ters 1307, base registers 1308, scientific registers 1309, T-registers 1310, status registers 1311, instruction counter IC 1312, and hardware control mask register 1313.

Typically the control store unit CSU 1301 is a 9K bipolar integrated circuit programmable read-only memory (PROM) mixed with read/write random access store (RAM). It has a typical 150 nanosecond read cycle and a 450 nanosecond write cycle. Each location of control store stores one 84-bit micro-instruction word (to be later more fully described), and each micro-instruction word controls one CPU cycle. As each location of the control store of the control store unit CSU 1301 is read, its contents are decoded by micro-operation decoders which provide micro-operation control signals each of which causes a specific operation within the CPU to take place (to be later described in detail).

By grouping locations within each micro-instruction word (to be later described in detail) control store sequences are obtained that can perform a specific CPU operation or instruction. As each instruction is initiated by the CPU, certain bits within the op-code are used to determine the control store starting sequence. Testing of certain flops (not shown) which are set or reset by instruction decode functions allows the control store memory to branch to a more specific sequence when necessary.

The control store interface adapter CIA 1302 communicates with the control store unit 1301, the data management unit DMU 1321, the address control unit ACU 1319, and the arithmetic logic unit ALU 1317 for directing the operation of the control store memory 1333 of Figure 13b. The CIA 1302 includes logic for control store address modification, testing, error checking, and hardware address generation. Hardware address generation is utilized generally for developing the starting address of error sequences or for the initialization sequence.

The data management unit DMU 1321 provides the interface between the CPU 1306 and the main memory and/or buffer store memory shown on Figure 1. It is the responsibility of the data management unit to recognize which unit contains the information required by other units and strobe the information into the CPU

registers at the proper time. The data management unit DMU also performs the masking during partial write operations.

The instruction fetch unit IFU 1318 interfaces with the DMU 1321, the ACU 1319, the ALU 1317, and the CSU 1301, and is responsible for keeping the CPU supplied with instructions. The instruction fetch unit has the next instruction available in its registers before the completion of the present instruction. To provide this capability, the instruction fetch unit IFU 1318 contains a 12-byte instruction register (not shown) that normally contains more than one instruction. In addition, the IFU, under control of the CSU, requests information (instructions) from main memory before the instruction is actually needed, thus keeping its 12-byte instruction register constantly updated. Instructions are thus prefetched by means of normally unused memory cycles. The instruction fetch unit also decodes each instruction and informs the other units of the instruction's length and format.

The address control unit ACU 1319 communicates with the IFU, ALU, DMU, and the CSU via the CIA. The ACU 1319 is responsible for all address development in the CPU. All operations of the ACU, including transfers to, from, and within the unit, are directed by CSU microoperation and logic in the unit. The normal cycling of the ACU depends on the types of addresses in the instruction rather than on the type of the instruction. Depending on the address types the ACU may perform different operations for each address in an instruction. The ACU also contains an associative memory 1319a that typically stores the base address of the 8 most recently used memory segments, along with their segment numbers. Each time a memory request is made, the segment number is checked against the associative memory contents to determine if the base address of the segment has already been developed and stored. If the base address is contained in the associative memory 1319a, this address is used in the absolute address development, and a considerable amount of time is saved. If the base address is not contained in the associative memory 1319a it is developed

by accessing the main memory tables. However, after the base address of the segment is developed, it is stored in the associative memory, along with the segment number, for future reference.

Interfacing with the ACU, IFU, DMU, and the CSU is the arithmetic and logic unit ALU 1317. Its primary function is to perform the arithmetic operations and data manipulations required of the CPU. The operations of the arithmetic logic unit are completely dependent on micro-operation control signals from the control store unit CSU 1301.

Associated with the ALU 1317 and the CSU 1301 is the scratch pad memory unit LSU 1315, (sometimes referred to also as the local store unit). It is typically comprised of 256-location (32 bits per location) solid state memory and selection and read/write logic for that memory. The scratch pad memory 1315 is used to store CPU control information and maintainability information. In addition, the scratch pad memory 1315 contains working locations which are primarily used for temporary storage of operands and partial results during data manipulation. Also associated with the ALU 1317 is an auxiliary memory 1317a comprised typically of 64 flip-flops for storing miscellaneous states of the computer system.

The CPU also has a clocking unit 1320 and is essentially 2 clocking systems in 1: the first clocking system generates the timing for the control interface adapter CIA 1302 and the second clocking system generates the timing pulses for the operations of the functional unit within the central processing unit.

Referring now to Figure 13c there is shown the format of the control store word 1325. The control store word is typically 84 bits wide and is divided into 6 main fields:

a. sequence type field 1326 (3 bits);

b. branching and/or micro-operations 1327 (23 bits);

c. constant generation and designation 1328 (14 bits);

d. data to bus 1329 (8 bits);

e. micro-operations 1330 (32 bits); and

f. checking 1331 (4 bits).

The 3-bit E field of the control store word 1325 is used as a sequence control field. There are typically 7 different sequence types and 1 reserved type for the instant computer system. Referring to block 1335 of Figure 13b, when E field equals binary 0, 1, or 2, the branching field A, B, C, D, and L of micro-instruction 1325 is utilized to generate the next address. The first 6 bits of KS register 1337 is utilized together with the B field, a C test results, the D test results and the L field to provide the next address of the next micro-instruction which is then placed in address register KS 1337. When the E field is set to binary 4 (see block 1335) the next address selected is taken from interrupt return register KA 1339. The address stored in the KA register is the one generated by the next address generation logic when the hardware interrupt occurs. When the E field is set to binary 5 a branch is used to initiate a subreturn from a microprogram subroutine. When used, the contents of the return register KR 1346 are used as the next control store address. The return register 1346 is loaded by issuing a control store command which will load present control store address in KS register 1337 plus 1, from incrementor 1338, into the KR register 1346. A one-level-nesting subroutine ability is provided via the KT return branch register 1347. Every time the KR register 1346 is loaded the old contents of the KR register is transferred to the KT register 1347 every time the microprogram return is called; the contents of the KT register will transfer to the KR register. Third level nesting subroutine ability is provided by the KU register 1340; and fourth level nesting subroutine ability is provided by the KV return branch register 1349. When the E field of the control store word is set to binary 6 the next control store word addressed is equal to the present address in KS register 1337 plus 1 in incrementor 1338. When the E field is set to binary 7 the CSU 1301 enters the diagnostic mode and the next address will be the present address plus 1.

In addition to the sequencing control of branching to the next control store address described above and shown in block 1335, there is hardware generated se-

quence control shown in block 1336 of Figure 13b. (Note: Blocks 1335 and 1336 are in actuality hardware registers drawn so as to depict the different forms that the micro-instruction words may take.) The hardware generated branches are overriding conditions (such as errors, initialize, control store scan, etc.) which surpass the E field and force a fixed address into the control store address register KS 1337. The branch is made by forcing an interrupt line high (not shown) for one clock period and storing the address which would have been generated under the control of the E field into the KA interrupt return register 1339. A hardware generated address will be placed into the control store address register. Certain hardware/firmware generated interrupts take priority when the interrupt-block flip-flop (not shown) which prevents additional interrupts in their class from being executed until the interrupting condition has been satisfied. A firmware micro-operation exists for controlling the resetting of the interrupt-block flip-flop for those sequences which are under firmware control. Those sequences under hardware control automatically generate a reset of the block-flop at the end of the sequence. The following conditions, listed by priority, exists in this category; (a) control store load; (b) control store scan; (c) hardware error; (d) software error. The remaining hardware conditions do not set the interrupt block-flop but do cause an immediate action to occur when generated. The following conditions listed by priority, exist in this category;

(a) initialize;

(b) soft-clear;

(c) enter maintenance panel;

(d) enter maintenance panel;

(e) hardware exit.

An initialize signal causes the CSU 1301 to branch to address binary 0, clear hardware resettable errors and execute a control store load operation followed by a control store scan sequence under hardware control. It will also perform system initialize. A soft-clear signal causes

the CSU 1301 to branch to address binary 0, clear hardware resettable errors and reset the interrupt block-flop. An enter maintenance panel signal causes the CSU to branch to the address preset in the CSU address switches on the maintenance panel (not shown).

An enter maintenance channel signal causes the CSU to branch to an address generated via the maintenance channel (not shown). The address loaded is from maintenance bus QMB 1344, which is part of the maintenance channel, and is right-justified. A hardware exit signal causes the CSU to branch to binary address 2. This sequence is used as a maintenance facility. At the end of the sequence a return is initiated by issuing an E field branch with the E field set to binary 4.

A control store load signal causes the CSU to branch to address binary 0. It also turns off the CSU read-cycle flop (not shown), the system clock 1320, and places the CSU in the load state. In the load state the CSU can be loaded from the control store loader CSL 1303, the IOC 1305, the main memory 102, or the maintenance panel 1355. When loaded from the CSL an automatic scan is generated at the end of the load. When loaded from any other media a scan may be issued by either generating a micro-operation signal or setting the scan switch on the maintenance panel. A control store scan signal causes the CSU to branch to an address binary 0. A control store scan is under hardware control for the duration of the sequence. During the scan the system clock 1320 is off and therefore no commands or tests are executed. At the end of the scan sequence the hardware transfers the contents of the interrupt return register KA to the address register KS, the system clock is turned on and control is returned to the firmware.

A hardware error signal causes the CSU to branch to address binary 4. In the normal processing mode a hardware error detected in any CPU functional unit will activate a hardware error line (not shown). The control store sequence generated will test the system conditions to determine the action to be taken. In the diagnostic mode, error conditions which are hardware detectable are visible to microdiagnostics. The microdiagnostics control

the action to be taken. A software error signal on the other hand causes the control store to branch to address binary 1. This address is the start of the software error reporting sequence which is under micro-program control.

Referring once again to Figure 13c the E field 1326 is a 3 bit field for the branch code as previously described. The branching and/or micro-operation field 1327 is comprised of the A, B, C, D, and L fields (also shown on block 1335 of Figure 13b) wherein the A field is the upper 6 bits of the next address, the B field is the middle 4 bits of next address of the mask field 64 tests, the D field is another 6 bit test field for 1 of 64 tests, and the L field is the least significant bit. The K field 1328 is a 14 bit field of which 6 bits are for the constant field, 4 bits are for a constant or steering field, and 4 bits are a steering field for a constant. The data to bus field 1329 is comprised of the QA field having 4 bits for controlling information to the QA portion of the QMB bus 1344 and the QB field has 4 bits for controlling information to the QB portion of the QMB bus 1344. The F field 1330 is a 32 bit field which is coded to generate micro-operation subcommands. The P field 1331 is comprised of 4 bits reserved for checking.

In operation the micro-instruction words are stored in the control store array 1333. During a cycle of operation, the control store array is addressed by the contents of the KS address register 1337. This causes the contents of the location specified by the address to be read into the group of read latches 1357. Portions of the word contents of the read latches are distributed or transferred to storage registers within each of the functional units in the CPU. Each functional unit includes decoding logic circuits for generating the requisite subcommands specified by the control store word under control of the system clock source. In general decoding is performed within each functional unit in the CPU rather than being performed centrally in order to minimize the decoding time and to reduce the number of cables which would be normally required for transmitting command signals if decoding were performed centrally. Additionally, the decoding is done within each unit to avoid timing problems

arising from differences in cable delays. Further, by decoding subcommands with each unit, those signals which are representative of certain conditions existing within the functional unit are required for the generation of certain subcommand signals do not have to be returned to the CIA unit 1302. A typical decoder unit 1359 is shown in Figure 13b as receiving various fields from micro-instruction words and generating micro-operation signals a, b, c, d, . . . q, r. A typical micro-operation decoder 1359 receives commands from a micro-instruction word. The field from the micro-instruction word is decoded and sets one of a plurality of lines s, t, u, . . . y, z high. A matrix is formed by having predetermined control line impedance coupled to the s-z lines at points α, β, γ . . . Ψ, ω. Typically then when the field from a micro-instruction is decoded one of the lines s-z goes high. Since the black dots shown in the matrix by Greek letters α through ω represent impedance coupling between the two sets of lines, any electrical signal propagating along any horizontal wire will be coupled through to propagate along the vertical wire a-r where an impedance coupling (black dot) is indicated. Each vertical line a-r then may be coupled as one input to one each of AND gates 1360-1365. Other input signals may also be coupled to AND gates 1360-1365. Other input signals may also be coupled to AND gates 1360-1365 including a timing signal t_s from the central timing unit. Accordingly as each timing signal t_s goes high those gates which have all the other input signals high will be enabled and will provide micro-instruction signals to predetermined functional units in the CPU. For example, if a command 1341 from read latch 1357 is decoded and a horizontal line is high the a, b, c, and q vertical control lines will be high and AND gates 1360, 1361, 1362, and 1364, will be enabled as the t_s timing signal is applied to these gates sequentially. Hence the combination in which the vertical control lines are coupled to the horizontal control line at different points represented by the Greek letters α through ω rep-

resent a permanent switching matrix for supplying micro-operation signals to the central processing unit CPU for controlling the functional units within the central processing unit by micro-instructions furnished from the control store array 1333. Thus permanent firmware having an alterability feature can be built into the machine of this invention by merely specifying the sequence of micro-operations that is required as a capability of the computer system.

Under normal conditions data is written into the control store array 1333 via the CPU write-data register also known as the local register YO 1343. A control flop (not shown) defines whether the uper half or lower half of the storage array is to be written. Data from the control and load unit CLU 1304 arrives at the CIA/CSU via the maintenance bus QMB 1344 and is buffered by the storage local register YO 1343 before being written into the control store array 1333. The storage local register 1343 is time shared as both a read and a write local register. The multiplexor KQM 1345 may be controlled by either the maintenance panel 1355 or by micro-diagnostics and provides a read out path from the registers connected to it. A compare register KP 1350 is provided for non-functional use and is used mainly for maintenance purposes and is utilized along with compare logic 1352 and decode logic 1351.

DISPATCHER FIRMWARE FOR CONTROL OF PROCESSES

The dispatcher is a firmware/hardware unit whose main purpose is to manage the various queues of processes, and to switch between processes and includes the updating of the queues of processes, the process control blocks PCB's, the running process word in the system base and the register of a new process. It also delivers messages to a process which is actually waiting for it on a semaphore (after a V-operation, a simulated V-operation for the IOC or for the exception handler). It also enqueues a message on a semaphore after a P-

operation which freed a message link when a process is waiting on the free-link semaphore to deliver its message.

The dispatcher unit further calls on the native mode instructions firmware after the "rolling-in" of a process executing in native mode, or after a "contest" if the current process remains running and is executing in native mode. It also calls on decor extension firmware for:

(a) a temporary call during the rolling-out of a process executing in that decor extension;

(b) a temporary call during the rolling-in of a process executing in that decor extension;

(c) a definitive call at the end of the rolling-in of a process executing in that decor extension; and,

(d) a definitive call after a contest if the current process remains running and is executing that decor extension.

Moreover the dispatcher places the system in the idle loop when there is no running process.

There are several ways of entering or leaving the dispatcher as follows:

(1) The initialization procedure (SIP) provides entry as the last step, (See Patent Application No. ———— entitled System Initialization Procedure, assigned to the same assignee named herein.)

(2) The Start and Suspend Instruction provides for the entry into the dispatcher. The Start instruction starts a process and the Suspend instruction terminates a process. (See Patent Application No. ———— entitled Start and Suspend Instruction.)

(3) The P & V operations provide for entry into the dispatcher. The P op picks up a message from a semaphore and if no message then the process goes into wait state. (See Patent Application No. ———— entitled "P and V Operations of Semaphores".)

Briefly then, the dispatcher is the main mechanism which manages the processes and accordingly the process control blocks PCB's by deciding which process is to be run and then taking the appropriate action, such as rolling-out the currently running process (i.e. write into the PCB all information relative to the currently running process contained in hardware registers, scratch pad memory, etc.), and rolling-in the new process (i.e. write

from the PCB to various hardware registers, scratch pad memory, etc., all information necessary to run the new process.)

The representation of the functions performed by the dispatcher are shown in flow-charts Figs. 14a-14i. For example, block 1402 of Fig. 14a is a representation of the function performed by the dispatcher wherein a micro-program word is delivered by the control store unit, and upon decoding by decoder 1359 controls the applicable portions of the CPU via an appropriate series of micro-operation signals 1360, 1361, etc. to retrieve the IPQW from the system base in storage subsystem 102, and transfer it to scratch pad memory 1315. Simultaneously the dispatcher fetches 1404 the GO segment descriptor (see Figure 12) from the G table of segment descriptors pointed to by the G table word GTW in the system base. Bits 16-31 of the IPQW word contain a 16-bit positive integer which is the displacement from the base of the G segment number 0, referred to as the GO segment, to the head (first byte) of the Q/PR/RDY, queue of processes ready. If bits 16-31 of the IPQW word is 0, 1403 the ready queue is considered to be empty. If the ready queue is empty it indicates that there is no process currently waiting in the Q/PR/RDY and the ready-queue is empty. The next question to be decided in decision block 1405 is whether or not a process is currently running in the machine by determining whether or not the Vacant Indicator is set. (The Vacant Indicator is a flip-flop located in auxiliary memory 1317a; it is set when there is no current process CJP running in the processor). If the Vacant Indicator is set (i.e. there is no process currently running) and since it was previously determined that there is no process in the ready-queue waiting to use the processor then the machine goes in the idle state 1406. However if there is a process currently running in the machine, but there is no one waiting to use the machine, the current process accesses its next instruction 1407.

Returning now to decision block 1403, of the flow chart of Fig. 14a if there is a positive integer in the pointer area of the IPQW (i.e. bits 16-31) the head of the ready

queue pointed to by the IPQW word in the GO segment is fetched to the scratch pad memory. (Note: To avoid repetition and in the interest of clarity the intermediate functions of the dispatcher in conjunction with the control unit and CPU will be omitted; however it is to be understood that such intermediate functions as previously described by example are present typically.) Up to this point it has been determined that there is some process waiting in the ready-queue. Before further action can be taken it is necessary to determine if there is a process currently running in the central processor. This is determined at decision block 1410 of the flow chart representation if there is no process currently running in the central processor (i.e. no CJP) the head of the ready queue is run 1412. However if there is a process running in the central processor the dispatcher must determine who has priority—the currently running process or the head of the ready queue? Accordingly therefore the priority byte of the current process (CJP) which is located in the running process word of the system base 600 or in process main word 0 PMW 0 of PCB 400, is fetched 1413. A decision is then made 1414 as to whether or not the current running process CJP is of lower priority than the new process NJP waiting at the head of the ready queue? (See decision block 1414.) If CJP is not of lower priority than NJP, CJP remains in control of the central processor and the contest indicator is reset 1415. (The contest indicator is always set to 0 except when one or more new processes have been placed on the ready-queue since the beginning of the last instruction executed on behalf of CJP thereby creating the possibility of a conflict; under these conditions the contest indicator is set to binary 1). Before the current process CJP is permitted to continue however, and execute further instructions a determination is made whether or not the CJP is running in decor extension mode 1415. If CJP is running in decor extension mode then the next instruction is executed in emulation mode (i.e. decor extension) and if it is not running in decor extension mode then the next instruction is executed in native mode. Returning once again to decision block 1414, if the NJP at

the head of the ready queue has a higher priority than the CJP (i.e. its priority number is lower than CJP's priority number, the current running process CJP is "rolled-out" of the machine and the new process NJP is "rolled-in" to the machine. Accordingly a firmware priority subroutine PRIQ 1418 directs the enqueueing of the current process CJP into the ready-queue by LIFO priority and by priority number by first "rolling-out" the CJP under the direction of the firmware subroutine RLLO 1419. The RLLO subroutine directs the writing of the information of the CJP that is stored in the general registers, base registers, scientific registers, T-register, status registers, and instruction counter, back into the appropriate storage areas of the process control block PCB in main memory, and it directs the updating of the RUA. Moreover the DEXT number of process main word 0 (PMW0) in PCB 400 is updated 1420.

The new process NJP is now ready to be "rolled-in". The boundary address register BAR is fetched 1422 and the running process word RPW is fetched from address BAR plus 56 of the system base. See block 1423. The name of the new process NJP is next written into the running process word RPW and since the name of the new process NJP was written in the process link PL of the Q/PR/RDY, the name in the process link PL is therefore now placed in the RPW, block 1424. Therefore NJP from the ready-queue now becomes CJP and is entitled to control the central processor and accordingly is no longer waiting in the Q/PR/RDY and must be dequeued by taking its name out of the process link PL of the Q/PR/RDY, block 1425. When this is done the queue of processes ready Q/PR/RDY is updated by the firmware subroutine UQLK, 1425a. Accordingly the JP number of the process which was just taken out of the machine is placed in a process link in the Q/PR/RDY, since it now no longer has control of the machine and must wait for it, 1426. At this point the transformation of giving this control of the central processor to the new process and placing the old process in a ready queue is accomplished and because there is a process (the new CJP) in control of the central processor, the Vacant

Indicator is set to 0, 1427; if on the other hand there were no CJP in control of the central processor the Vacant Indicator would be set to 1. At this point the assignment of the processor is complete and a new process has acquired the central processor whereas the old process was placed in a ready queue; however, the new process is not ready to run yet for the hardware of the central processing unit, 1306 of Figure 13a, such as the general registers 1307, the base registers 1308, scientific registers 1309, the T-register 1310, the status registers 1311, and the instruction counter 1312, must be supplied with control information from the process control block of the new process.

Accordingly the firmware subroutine 1430 first fetches PMW 3 from the PCB (Figure 4) to scratch pad memory 1315, and then fetches PMW 0. The MBZ field of PMW 0 is checked 1433 and if it is not binary 0 an illegal PCB exception results. However if the MBZ field of PMW 0 is zero then PMW 1 is fetched 1434. Again the MBZ field of PMW 1 is tested to determine whether or not it is a binary 0. If it is not binary 0 there is an illegal PCB exception; whereas if it is equal to 0 the dispatcher proceeds to C.

Accordingly address space word 0 ASW 0 is fetched from the appropriate space in the PCB and the segment table word sixe STWSZ is tested 1437 to determine whether or not it is less than 7. If it is greater than 7 an illegal PCB results if it is less than or equal to 7, then ASW 1 is fetched from the PCB, block 1438, and its STWSZ field is tested 1439 to determine whether or not it is less than or equal to 8. If its field is greater than 8 then an illegal PCB results; however, if its STWSZ field is equal or less than 8 the exception word EXW is fetched 1440, and its MBZ field is tested to determine whether or not it is equal to 0. If its MBZ field is not equal to 0 an illegal PCB results whereas if it is equal to 0 the stack word SKW is fetched 1442, and its MBZ field is tested 1443 to determine whether or not it is equal to 0. If the MBZ field is not equal to 0 then an illegal PCB results; whereas if it is equal to 0 the instruction counter word ICW is fetched from the PCB and placed

in the instruction counter IC and its TAG field is tested to determine whether or not it is equal to 0, 1445. If the TAG field is not equal to 0 then an illegal PCB results. However if the TAG field is equal to 0 the MBZ word is fetched 1446 and its MBZ field (bits 0-31) is tested to determine whether or not it is equal to 0, 1447. If it is not equal to 0 then an illegal PCB results; whereas if it is equal to 0 stack base words 0, 1 and 2 SBW 0, 1 and 2 are fetched 1448. The contents of the 8 base registers in the base register saving area of the PCB is then fetched 1449 and stored in the machine's base registers 1308. Then the contents of the 16 general registers from the general register saving area of the PCB are fetched 1450 and stored in the machine's general registers 1307. Before fetching the contents of the scientific registers however a check is made of the capability byte of process main word 0 (PMW 0) to determine whether or not scientific mode is being utilized 1451. If scientific mode is being utilized then the contents of the scientific registers from the scientific register saving area of the PCB are fetched and stored 1452. The firmware then proceeds to make a check of the capability byte of PMW 0 to determine whether or not the accounting mode is being utilized 1453. If the accounting mode is being utilized (i.e. accounting bit of capability byte set to binary 1) the accounting words exists in the PCB and the ready time account word RTA is updated. Then the firmware proceeds to determine whether or not the DEXT number is set to 0, 1454. If it is not set to 0 it indicates that the machine may be in emulation mode (i.e. the decor extension capability is being utilized) and accordingly the DEXT number of PMW 0 is checked 1455 to determine whether or not it is greater or less than the DETSZ field of process main word 3 and if it is greater than the DETSZ field an illegal PCB exception 1456 results since the DEXT number is less than the DETSZ field but not equal to zero the machine is performing in legal emulation mode and proceeds to F. Returning to decision block 1454 if the DEXT field is binary 0 then native mode is being performed and the machine fetches STW's 1457. The residual time out word RTO word of the PCB is

fetched 1458 and the process timer is loaded, with the time that the CJP may spend in the running state.

Up to this point either (a) a new process NJP has been "rolled-in" to take control of the CPU when there was an old process CJP in the machine and the new process NJP was at a higher priority than the old process CJP or (b) there was no CJP in control of the CPU and the head of the ready queue was rung. Briefly under condition (a) the CJP was taken out of the RPW and placed in a process link PL in the Q/PR/RDY, and the NJP in a process link PL in Q/PR/RDY was placed in the RPW thus effectively switching the positions of the 2 processes giving control to the NJP which now becomes the CJP and taking control away from the old CJP. Then the PCB of the NJP was accessed and the information required to run NJP (now the CJP) was placed in scratch pad memory or the array of registers in the ACU.

If there was no CJP in control of the CPU (condition b), then the head of the ready queue was run—i.e. the NJP became the CJP because of the dispatcher taking the NJP from the process link PL from the head of the ready queue and placing it in the RPW word. By doing this a process link PL was left empty in the Q/PR/RDY and it is required to take it out.

Accordingly beginning now at decision block 1461 the firmware determines whether or not there was a CJP in control of the CPU and if there was a free process link (FPLS) it was accessed and enqueued and the CJP was written therein. However if there was no CJP in control of the CPU state byte of PMW 0 of the NJP is updated 1460 and again there is a determination whether or not there was a CJP in the machine 1463. If there was no CJP in control of the processor the process link of the NJP (which was in the Q/PR/RDY and is now in control of the machine) is taken out of the Q/PR/RDY 1466 (i.e. dequeued from the Q/PR/RDY) and becomes a free link semaphore FLSP and is now enqueued in the free process link queue (805 on Figure 9) and becomes a part of the free process link queue 1466a. The contents of the boundary address register BAR are fetched 1464 and the

running process word RPW of the NJP (now the CJP) located at BAR plus 56 of the system base is updated by placing the NJP identification in the RPW, block 1465. The Vacant Indicator is set to 0 in the case where there had been no CJP. Next the Contest Indicator (i.e. a flip-flop in auxiliary memory 1317a used to indicate a possible conflict in priority between the CJP and a process placed in the ready queue) is set to 0, 1467 and the Segment Associator (AS 132 on Figure 1) which is a typical content addressable memory is cleared, 1471; then the process mode is entered 1470. (A process mode indicates that exceptions are handled by the process which is operating in the processor rather than by the operating system). The firmware then continues to CAB 1480 and the asynchronous trap bit AB is checked to determine whether or not it is set to binary 1, 1481. If the AB bit is set to binary 1 then a check is made 1482 to determine whether or not the process ring number PRN is greater than or equal to the asynchronous trap bit ARN. (The AB and the ARN are located in the priority byte of the PCB of each process and is meaningful when the process is in the running state; the AB and ARN are obtained from the RPW located at BAR plus 56 of the system base). The AB and ARN in BAR plus 56 of the RPW are reset since the next step 1484 is to proceed to an asynchronous trap routine which would take care of the conditions that cause the asynchronous trap bit or the asynchronous ring number to be set in the first place, and if these were not reset then at the next pass by the firmware it would give an indication that something was wrong where actually there was nothing wrong, and hence would always proceed to the asynchronous trap routine 1484 and would never execute. Going back now to decision blocks 1481 and 1482 if the AB bit is not set or AB bit is set and PRN is not greater than ARN, then the firmware would proceed to determine in what mode the processor would run—in the normal mode or in the emulation mode. Accordingly the DEXT number is checked to determine whether or not it is set to 0 and if it is set to 0 the normal mode of the machine is run 1487. However if the DEXT number is not set to 0 the emulation mode is run 1486.

The Switch System Base Instruction

Referring now to Figure 15a there is shown the structure of the switch system base instruction. The operation code OP 1501 is similar to the operation code of any instruction and indicates the operation to be performed. The complementary code C 1502 is used in conjunction with the op-code 1501 when the op-code shares more than one instruction 1501 and the complementary code 1502 determines the precise instruction to be performed. For example, the P and V operations have 12 instructions and they will use one op-code. The complementary code together with the op-code determine which of the 12 P and V instructions is to be performed. The address syllable AS 1503 of the switch base instruction is the address pointing to a new copy of the system base which is to be loaded into main memory and also scratchpad memory. Accordingly, this address points to an entirely new system base. Typically a 60 byte zone, starting at the address specified by the address field of the switch system base instruction, is transferred into the system base area which starts at address BAR. This transfer is also made into the scratchpad memory. This zone, therefore, is the new system base which replaces the previous system base. The running process, therefore, disappears because its PCB is not updated and no link manipulation is performed. The processor is then placed in the VACANT state, and the dispatcher entered.

It is the system software responsibility to prepare the new information for the system base and the new GO segment before executing the SWSB instruction. (See applications included by reference herein numbers 12-16). For that reason, the SWSB instruction must be entered with the hardware gate HG locked, otherwise it returns a condition code CC without taking further action. Upon completion of SWSB instruction the HG will be left unlocked. Also upon completion of the SWSB instruction the vacant switch is set to 1. This indicates no process is running and as a result the system will go to the head of the ready queue for the internal process queue word IPQW and pick out the pointer to the queue of processes ready. It will then proceed to the head of

the ready queue and start up a new process. The process which executed the SWSB instruction will terminate because a vacant condition has been placed in the vacant indicator, which means that there is no running process.

Referring now to Figures 15b and 15c the SWSB instruction in firmware will be described in greater detail. The SWSB instruction starts 1504 and a determination is made 1505 whether or not the process ring number PRN is 0. This is necessary so that the SWSB instruction executes with the highest privilege. If the PRN is not 0 then a privileged instruction exception 1506 occurs. If on the other hand PRN is equal to zero the hardware gate located in auxiliary memory 1317a of the ALU 1317, is checked 1507 to determine whether or not it is locked. It is important that the HG is locked during the execution of the SWSB instruction. When the HG is locked no I/O interrupts or exceptions are permitted. If such exception or an error condition occurs while the HG is locked the system will crash. The system crash is a more desirable alternative while the system base is being changed, then permitting the interrupt with the system base only half changed. Accordingly, if the HG is not locked the condition code CC in one of the status registers is set to 2, (block 1508) which in this case is the code indicating the hardware gate HG is not locked.

Accordingly, the SWSB instruction is not allowed to proceed 1509, until the hardware gate HG is locked. If on the other hand the hardware gate HG is locked a determination is made as to whether or not the new system base word (i.e. address syllable AS) is quad-word aligned. If the AS is not quad-word aligned then the condition code CC is set to 3 indicating this condition and the SWSB instruction terminates 1509. On the other hand if the AS is quad-word aligned the condition code CC 1512 is set to 0 which indicates that the SWSB instruction was successful and can proceed to perform the move operations 1513 from the new system base image into the system base area located at BAR. Finally after the new system base image has been transferred into the BAR system base the hardware gate is unlocked 1514,

the vacant indicator in the auxiliary memory 1317a is set to 1, (block 1515) indicating no running process, and finally the instruction is complete 1516.

Figure 15c is a more detailed flow diagram of block 1513 which is the step that transfers the new system base image to the BAR system base area, and includes loading of the scratchpad memory S.P.M. with information from new system base image.

Referring to Figure 15c the start 1517 of SWSB instruction is actually at block 1513 of 15b. Typically 60 bytes of the new system base image located at address AS are transferred 1518 to the system base whose address is indicated by the boundary address register BAR. A new G table word GTW from the new system base image is loaded 1519 into scratchpad memory S.P.M. in auxiliary memory 1317a. Then a new J table word JTW is loaded 1520 in S.P.M. Finally a new internal processor queue word IPQW is loaded 1521 into S.P.M., and that portion of the instruction completes 1516 and continues to block 1514, where the operations are as previously described.

GLOSSARY OF TERMS

Absolute Address —the physical address of a hardware byte in main memory store.

Address Development —a hardware function which operates on a number of address elements to compute an absolute address which is used to refer to a location in main storage.

Addressing —locating an object by any of a number of virtual, logical, and physical means.

Address Space —the set of logical, segmented addresses corresponding to a process that the CPU is permitted to transform into absolute addresses during execution of that process.

Address Space Word —one of two words in a process control block which point to the segment table word array. The segment table word array defines the segment tables associated with the process.

Address Syllable —a logical address recognized by CPU hardware, normally an operand of an instruction.

Analyzer/Translator —that part of the static linker that first passes the control language stream to the linker, then, if the control language is error-free, the analyzer/translator converts it to tables and other structures for the static linker to use.

Asynchronous Processing —concurrent activation of more than one process.

Auxiliary Memory —is comprised of 64 flip-flops for storing miscellaneous states of the system and is located in ALU.

Base Register —the primary element in segment addressing, referenced by number in every address syllable.

Boundary Address Register —a visible hardware register that defines the lowest memory address accessible to firmware/software.

Call —see procedure call—Inward Call-Outward Call.

Central Process Unit —a part of a computer that includes circuits controlling the interpretation and execution of instructions.

Channel —a means of communicating between the processor subsystem and a peripheral subsystem. There are two types of channels, physical and logical. A physical channel is the hardware connection between the input/output control unit IOC and the peripheral control

unit PCU. A logical channel is a software communications path between main memory and a single peripheral device.

Channel Command Entry —an instruction in a channel program.

Channel Command Word —an element of a channel command entry. Two channel command words constitute a channel command entry.

Channel Program —a sequence of instructions that cause a specific input/output i/o operation to be performed by a peripheral device.

Complementary Code —a code in some instructions which together with the operation code defines the function of the instruction.

Compilation Unit —the single object-code module that results from compiling or assembling one procedure in any high level language processor. The CU is not executable unit it is linked.

Concurrency —apparent simultaneity.

Condition Field —a 4-bit field in a branch instruction. Each bit of the field corresponds to a setting of the condition code in the status register bits 0 and 1.

Contest Indicator —located in auxiliary memory and indicates a new entry was placed in the ready queue.

Data Address Space —the set of logical record addresses accessible by a process through data management; consists of four elements; logical fields, logical records, data base pages, and files.

Data Descriptor —an information structure which is used in indirect addressing and as a description in data manipulation.

Decor —the functional characteristics or architecture of a computer system.

Decor Extension Mode—a mode of operation permitting the operating of the system in emulation mode.

Descriptor —a 32- or 64-bit field used in developing the address of any data field. It is one of many pointers in the addressing scheme of the invention.

Descriptor Block —a data block containing any number and variety of descriptor declarations except semaphore descriptors. No other data is allowed.

Device Adapter —electronics which adapts a specific device to a peripheral controller.

Device Adapter Interface —the interface between a device adapter and its associated peripheral controller(s).

Device Control —a software facility which handles device specific characteristics.

Dispatcher —the firmware responsible for dispatching processes in the CPU.

Dispatching —a field within the instruction format or a data descriptor which provides relative position information within a segment.

Displacement Addressing —a form of direct addressing in which the operand defines a location relative either to the base of the immediate segment or the current ation counter.

Dynamic Addressing —addressing which results in resolution, during execution, of references defined within the process group.

Editor —that part of the static linker that format and outputs any or all of the information that results from linking a particular procedure.

Effective Address —a logical, as opposed to physical, address consisting of a segment table number, a segment table entry, and a segment relative address. These elements show the way to a particular segment descriptor, which eventually points to the segment being referenced.

Emulation Mode —operation of a computer so as to initiate another computer, and appear to operate in the other computer's environment.

Exception —an exception occurs when the hardware detects the existence of some condition that has been specified as requiring special handling.

Event —anything observed in a system by a process which may be of interest to some other process.

Firmware —that part of a hardware unit implementation which utilizes a form of micro-programmed control.

G, D Name —the name used by the hardware to refer to a semaphore. G, the G-segment name is the number of the entry in the G-table containing the segment descriptor, D is the relative address in the segment.

G-Segment —a segment containing semaphores which may be addressed using the G, D name form and associated mapping facility (the G-table).

G-Table —a table used for storage mapping (locating a G-segment) when the G, D name form is used.

Gating —a facility for controlling entry to procedures in a (gated) segment through the use of procedure descriptors.

General Register —a 32-bit register available to executing processes which generally holds binary or bit string data. Certain general registers can be used for indexing (GR8 through GR15).

Index —a data structure which is maintained and operated upon by the system. The structure is visible to the user only in terms of a key which he presents for the purpose of making available or positioning to a specific record occurrence.

Indexing —modification of an address by a signed arithmetic value by adding something to something.

Index Register —a general register used for indexing (GR8 through GR15).

Indirection —a method of addressing whereby an address is used to obtain another address, rather than the actual data.

Indirection to Base —the form of indirection whereby a data descriptor containing a base register number and a displacement is found at the address referenced.

Indirection to Segment —the form of indirection whereby a data descriptor containing a segment address is found at the address referenced.

Indirect Segment Descriptor —a type of segment descriptor which contains information for addressing another segment descriptor, rather than a segment.

Instruction —the unit of execution of a central processor visible to a programmer.

Instruction Counter	—a register associated with an executing process which contains the segmented address of the next instruction of the procedure being executed.
Interleave	—sequential accessing of memory modules to reduce memory access time.
Interrupt	—the discontinuance of a process in the system due to an event occurrence.
Interrupt Process	—a process which is activated in response to an interrupt.
Inward Call	—when one procedure execution in one ring number calls another procedure to execute in a lower ring number.
I/O Controller	—a unit which provides the basic control for a specific I/O subsystem.
I/O Processor	—potentially asynchronous system processes which deal primarily with the movement of data between peripheral storage or I/O devices and main storage.
J.P. Tables	—a collection of logical addresses for locating a process control block.
Job	—a unit of work for the system.
Job Control Language	—a language for describing a job and its component parts.
Job Step	—the major subdivision of a job leading to the execution of a program defined by a JCL statement.
Job Step Table	—a system table used to map the first part of a process name, i.e., to locate the process group table.
Linked Module	—the output of the static linker. It is a consolidated group of compilation units (CU) whose cross-references were resolved by the static linker.

Linking	—see static linking.
Logical Channel	—see channel.
Logical Processor	—the collection of hardware resources and control information necessary for the execution of a process.
Main Storage	—all addressable storage from which instructions can be executed or from which data can be loaded directly into registers.
Magnetic Tape Controller	—the elements of the peripheral subsystem related to magnetic tape devices.
Mass Storage Controller	—the elements of the peripheral subsystem related to mass storage devices.
Memory Management	—operating system facilities for the allocation, relocation, and disallocation of physical memory.
Message	—information transmitted from a source or to a destination—neither being a file.
Microinstruction	—same as microinstruction word, and control store word.
Micro-program	—a set of machine codes used to implement the control functions of a processor.
Multiplex	—to share a hard resource, such as memory, usually by time division.
Multi-programming	—the concurrent execution of two or more programs by a single computer.
Native Mode	—operation of a computer in its native environment.
Offset	—in address development, the number of bytes after the beginning of the segment at which an addressed part of the segment starts.

Operating System	—a system of basic software designed to support the efficient operation of user software.
Outward Call	—when one procedure executing in one ring number calls another procedure to execute in a higher ring number.
Peripheral Control Unit	—a stand-alone micro-programmed processor which executes channel programs to perform I/O operations.
Peripheral Subsystem Interface	—a standard interface used for transfer and control between freestanding peripheral control units and I/O control units.
Physical Channel	—see channel.
Physical Data Structure	—a complete definition of the organization of data as it is physically recorded on media.
Physical I/O	—that part of the operating system which initiates and controls the transfer of data between memory and peripheral or terminal devices.
Physical Storage	—hardware used for the storage of data. It is composed of various types of recording media and the recording/reading hardware.
P-operation	—a machine instruction involving a semaphore which causes a process to enter the wait state or to receive a message.
Privileged Instruction	—an instruction which must be executed in ring 0.
Procedure	—a named software function or algorithm which is executable by a processor without concurrency: a Fortran subroutine, a Cobol program, a PL/1 internal procedure.

Procedure Call	—a mechanism for creating a stack frame within a stack segment.
Procedure Descriptor	—a word containing the location and entry point of a procedure.
Procedure Segment	—the type of segment whose contents represents a procedure, procedures, or part of a procedure.
Process	—the ordered execution of instructions without concurrency by a processor, either central or i/o.
Process Address Space	—the memory addresses which may be referenced or to which control may be transferred during a particular process.
Process Control Block	—a data structure (hardware defined and recognizable) that contains information needed to specify the state of a process at any given time.
Process Group	—a related set of processes, usually those necessary for the performance of a single job-step.
Process Group Builder	—that part of the static linker whose actions yield the linked module.
Process Group Loader	—an operating system facility that makes final resolutions in a linked module to create an executable process group then it loads the process group into memory and starts its execution.
Process Group Table	—see P Table.
Process Link	—an entry in a ready process or waiting process queue.
Process State	—the dynamic condition of a CPU process e.g., running, ready, waiting, or suspended.
Process Switch	—the hardware function that disassociates one process from the central processor and associates another.

Process Synchronization —the functions that act to coordinate the asynchronous activity among processes. These functions generally involve semaphores and the P and V operations.

Processor —a unit capable of receiving data, manipulating it, supplying results, and controlling its sequence of operations in response to stored programs: short for central processor, generic for central processor, peripheral processor, or hardware/software processor.

Program —the specification of the procedural and associated information necessary to solve a problem, i.e. the ordered collection of instructions which is executed by the computer and operates on data to perform a user's job or some phase of that job.

Program Execution —the activity of a process in conformance to program specification.

Protection —the function, enforced by hardware and software, of preventing processes from interferring with each other or sharing each other's address space in an unauthorized manner.

P-Table —a hardware defined data structure containing entries, each pointing to the origin of a process control block. The P-table entries point to the total set of PCB's comprising a process group at any given time. Also called process group table.

Q/PR/RDY —queue of processes ready i.e. a ready-queue.

Queue —an ordered list of entities awaiting information, process or availability.

Ready State	—a process state in which no processor is associated but all necessary resources, except a processor, are available for changing to the running state.
Relative Address	—the incremental/decremental location of some object relative to another.
Relocation	—movement of a segment from one location in main storage to another including the adjustment of all required references to its previous location.
Return	—the function and operations necessary to resume the action of a procedure at a point immediately following the point where it entered some other procedure.
Ring	—a protection attribute of a segment that restricts the read, write, and execute access of process to that segment. A level of privilege of a process for read, write or execute.
Rolling-in	—performing the operation necessary to give control of a processor to a new process.
Rolling-out	—performing the operations necessary to take away control of a processor by a process.
Running State	—the dynamic condition of a process (connected to a CPU) that is executing.
Scratch Pad Memory	—256 locations in ALU for storing CPU control information-also known as a local store unit.
Scheduling	—determining the sequence of operations.
Scientific Registers	—a 64-bit register used in the manipulation of binary floating point numbers. There are four scientific registers in the invention when the system has the scientific option.

Segment	—a contiguous main memory space treated as a unit.
Segmentation	—the use of a division of main memory into logical groupings called segments rather than into a single linear store.
Segment Base	—the current origin of the segment. This is one of the fields in a segment descriptor word.
Segment Descriptor	—entry in a segment table defining the characteristics of a segment or pointing to a segment descriptor.
Segmented Address	—see effective address.
Segment Number	—the identification of a particular segment consisting of a segment table number (selecting one of the segment tables of the process) and a segment table entry of that chosen table.
Segment Relative Address	—during address development, the final value to be added to the segment base to obtain the absolute address.
Segment Table	—a table containing segment descriptors used to establish the addressability of a process to its segments. The collection of segment tables of a process defines the address space for that process.
Segment Table Entry	—an item in a segment table. Each segment table entry is a segment descriptor.
Segment Table Word	—an entry in a segment table word array pointing to the segment tables of a process. The array is pointed to by one of two address space words in the process control block.
Semaphore	—data structures used to control communication between processes.
Semaphore Block	—a data block containing only semaphore descriptor declarations.

Stack —a mechanism that accepts stores, and allows retrieval of data on a last-in-first-out basis. Comprised of a number of contiguous portions called stack frames.

Stack Active Area —the portion of the current stack frame that holds a pointer to the saving area, the caller's instruction counter, parameter space and local variables—in short, the data of the current frame most likely to be useful to execution of the called procedure.

Stack Base Words —a group of three words in a process control block which contain the segmented address of the stack segments for rings 0, 1, and 2 associated with the process.

Stack Frame —a contiguous part of the stack segment holding the data stored for a single procedure call.

Stack Operations —the push-down and pop-up operations of storing and retrieving information to/from the stack. In practice, manipulation of the pointers used to reference the information stored in the stack.

Stack Overflow —the action of attempting to store more information into the stack then the available space can hold. This is detected by hardware.

Stack Register —see T register.

Stack Segment —the segment of a process that contains its stack.

Static Linking —an intermediate step between compiling a source procedure and executing the object form of that procedure; linking resolves the external references made to and from that procedure.

Status Register —an 8-bit register which specifies the status of the procedure currently being executed.

Swapping —releasing the main memory space occupied by a segment by writing the segment out to secondary storage. The actual write to secondary storage may be suppressed if the segment has not been modified since the last time it was read in.

Synchronization —having a fixed relationship in time; synchronized execution of two or more processes.

System Base —a fixed area in main memory which acts as a root for all information structures.

Task —the smallest unit of user-defined work consisting only of a single stream of non-concurrent instructions.

T Register —a software visible hardware register which contains the segmented address of the top of the procedure stack.

Unit Record Controller —the elements of the peripheral subsystem related to unit record devices such as card equipment, paper tape equipment, and line printers.

User Process Group —the internal representation of any job step, as opposed to a system process group which exists independently of any job.

Vacant Indicator —a hardware structure that indicates that the process which had control of the CP has suspended itself i.e. no CJP in the CPU. Located in Auxiliary Register, of the ALU.

Virtual Memory —an addressing concept that allows a programmer to code without regard to physical memory size; virtual memory manager automatically manages the exchange of segments between main memory and secondary storage.

V-Operation —a machine instruction intended for interprocess signaling the completion of an event. A V-OP operates on a designated semaphore.

ABBREVIATIONS

ACU —address control unit
ALU —arithmetic and logic unit
AS —address syllable
ASW —address space word
BAR —boundary address register
BCD —binary coded decimal
BR —base register
CCE —channel command entry
CCU —channel control unit
CMRN —the caller maximum ring number
CCW —channel command word
CET —current state entry time
CIA —control store interface adapter
CJP —currently executing process
CPU —central process unit
CU —compilation unit
CSU —control store unit
D —displacement
DMU —data management unit
DA —device adapter
EAR —effective address ring
EXW —exception word
GR —general register
GTW —G-table word
IC —instruction counter
IFU —instruction fetch unit
I/O —input/output

IOC —input/output controller
IPQW —internal process queue word
IR —index register
ITBB —indirect to base register
ITS —indirect to segment
J —job
JCL —job control language
JTW —J-table word
LCT —logical channel table
LIFO —last-in-first-out
LSU —local store memory or scratch pad memory
MBZ —must be zero
MOS —metal oxide semiconductor
MAXR —the maximum ring number at which a procedure may execute MAXR is found in the segment descriptor of SEGEP.
MSC —mass storage controller
MTC —magnetic tape controller
NFS —non-functional status
NPRN —new process ring number
NJP —new process replacing the currently executing process
PCB —process control block
PCT —physical channel table
PCU —peripheral control unit
PL/I —programming language/one
PMW —process main word
Q/PR/RDY —queue of processes ready
RD —read ring
RHU —reserved for hardware use
RPW —running process word
RSU —reserved for software use
RTA —ready time accounting
RTO —residual time out
SBW —stack base word
SEG —segment number
SKW —stack word
SEGEP —the segment which contains the entry point as found in the procedure descriptor
SEGPD —the segment which contains the procedure descriptor

SR	—scientific register
SRA	—segment relative address
STE	—segment table entry
STN	—segment table number
STR	—status register
STW	—segment table word
STWA	—segment table word array
T	—T-register (stack register)
URC	—unit record controller
WR	—write ring
WTA	—waiting time accounting

Having shown and described one embodiment of the invention, those skilled in the art will realize that many variations and modifications can be made to produce the described invention and still be within the spirit and scope of the claimed invention.

What is claimed is:

* * *

For Claims 3, 5 and 6, see Rejected Claims 3, 5, and 6.

DECLARATION AND POWER OF ATTORNEY — (for use with sole or joint patent applications accompanying or not accompanying the application papers)

As a below named inventor, I hereby declare that the information given below is true. That I believe that I am the original first and sole inventor of the invention described and claimed in the specification X attached hereto or my/our patent Application Serial Number _____ filed _____ that as to common subject matter of this application and any earlier filed United States application of any above named to which I claim priority below, I do not believe that the same was ever known or used in the United States before my/our invention thereof or more than one year prior to such earlier application, or on public use or on sale in the United States more than one year prior to such earlier application(s), that the said common subject matter has not been patented before the date of such earlier application(s) in any country foreign to the United States on an application filed by me/us or my/our legal representatives or assigns more than twelve months prior to such earlier application(s) and that no application for patent on said common subject matter has been filed by me/us or my/our legal representatives or assigns in any country foreign to the United States prior to this application except those identified in item(s) below. If any, that as to any subject matter of this application which is not common to any said earlier application(s) (hereinafter subject matter), I am the first inventor, and do not believe that the same was ever known or used in the United States before my/our invention thereof or patented or described in any printed publication in any country before my/our invention thereof or more than one year prior to the date of this application, or on public use or on sale in the United States more than one year prior to the date of this application, and that said non-common subject matter has not been patented in any country foreign to the United States on an application filed by me/us or my/our legal representatives or assigns more than twelve months prior to the date of this application, and that no application for patent on said non-common subject matter has been filed by me/us or my/our legal representatives or assigns in any country foreign to the United States prior to this application except those identified in said item(s) below.

FULL NAME OF APPLICANT (INCL. AT LEAST ONE UNABBREVIATED FIRST OR MIDDLE NAME)

	LAST	FIRST	MIDDLE
2	Bradley	John	J.

RESIDENCE

	CITY	STATE (OR FOREIGN COUNTRY)	COUNTRY OF CITIZENSHIP
0	Garches	France	U.S.A.

POST OFFICE ADDRESS

	STREET ADDRESS	CITY	STATE (OR FOREIGN COUNTRY)	ZIPCODE
1	63 rue du 19 Janvier 92380	Garches	France	

FULL NAME OF APPLICANT (INCL. AT LEAST ONE UNABBREVIATED FIRST OR MIDDLE NAME)

	LAST	FIRST	MIDDLE
2	Franklin	Benjamin	S.

RESIDENCE

	CITY	STATE (OR FOREIGN COUNTRY)	COUNTRY OF CITIZENSHIP
0	Cambridge	Massachusetts	U.S.A.

POST OFFICE ADDRESS

	STREET ADDRESS	CITY	STATE (OR FOREIGN COUNTRY)	ZIPCODE
2	11 Peabody Terrace Apt. #2001	Cambridge	Massachusetts	02138

FULL NAME OF APPLICANT (INCL. AT LEAST ONE UNABBREVIATED FIRST OR MIDDLE NAME)

	LAST	FIRST	MIDDLE
2			

RESIDENCE

	CITY	STATE (OR FOREIGN COUNTRY)	COUNTRY OF CITIZENSHIP
0			

POST OFFICE ADDRESS

	STREET ADDRESS	CITY	STATE (OR FOREIGN COUNTRY)	ZIPCODE
3				

LISTING OF APPLICANTS CONTINUED ON PAGE 2 (if any) — YES X NO

TITLE OF INVENTION

3 8 1 — Switch System Base Mechanism

Earliest foreign application, if any, filed within 12 months prior to the U.S. filing date of this application of which priority is claimed under 35 USC 119

☐ YES ☐ NO

COUNTRY	APP. NO.	DATE OF FOREIGN FILING	CONTINUED ON PAGE 2 HEREOF
			YES X NO

☐ Specific reference to any prior related earlier filed U.S. application(s) of which this application is a CONTINUATION / CONTINUATION-IN-PART / DIVISION is made on page ____ of the specification for the purpose of receiving benefit of said earlier filing date(s) 35 USC 120

I/We hereby appoint the following as my/our X attorney(s) agent(s) to prosecute this application and transact all business in the Patent Office connected therewith

NAME	REGISTRATION NUMBER	ASSOCIATE POWER OF ATTORNEY OR AUTHORIZATION OF AGENT ATTACHED
Nicholas Prasinos Ronald T. Reiling	24,791 20,607	YES X NO

SEND CORRESPONDENCE TO

NAME	STREET	CITY AND STATE	ZIP CODE	TEL. NO.
Ronald T. Reiling Honeywell Information Systems Inc.	200 Smith Street	Waltham, Massachusetts	02154	(617) 890-8400, Ext. 42126

I/We further declare that all statements made herein of my/our own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon

SIGNATURE(S) OF APPLICANT(S)	X HEREIN *	OR PAGE 2 ATTACHED *	DATE OF SIGNATURE
John J. Bradley Signature of Inventor			8 April 75 Date

* (IF PAGE 2 ATTACHED, SIGNED AND MADE A PART HEREOF) YES X NO

* (IF NOT SIGNED ON THIS PAGE, THIS DECLARATION IS NOT VALID UNLESS PAGE 2 IS SIGNED)

DRAWING
WAS ORIGINALLY FILED

570331

FIG. 1

S?0331

PRINT OF DRAWING
AS ORIGINALLY FILED

FIG. 2

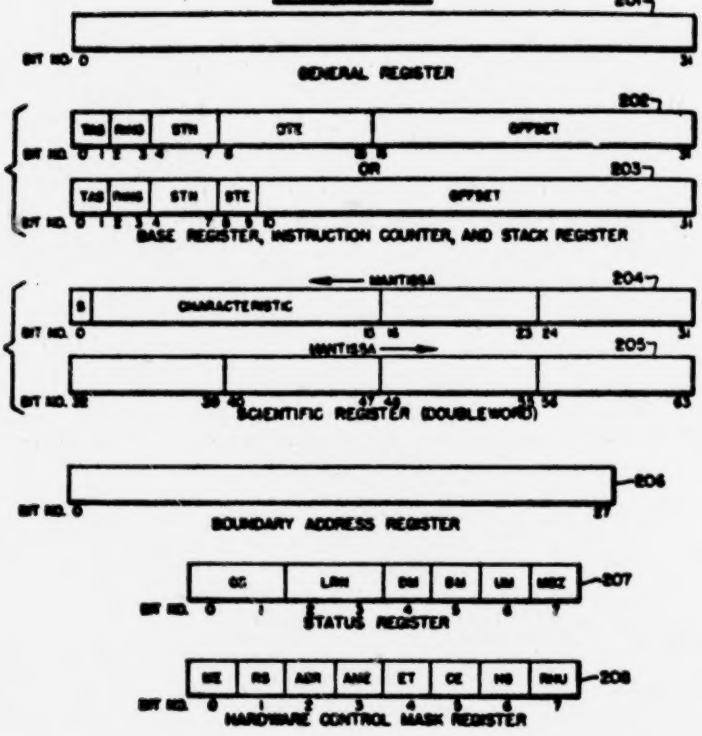


FIG. 3

LEGEND FOR RESERVED AREAS OF STORAGE IN REGISTERS

AME • ACCOUNTING MECHANISM
ABR • AUTO STORAGE RECONFIGURATION
BM • BINARY MASK
CC • CONDITION CODE
CE • CORRECTED ERROR
DM • DECIMAL MASK
ET • ENVIRONMENT TOLERANCE
HS • HARDWARE GATE
LRN • LAST RING NUMBER
MBZ • MUST BE ZERO
ME • MACHINE ERROR
RING • RING NUMBER
RHU • RESERVED FOR HARDWARE USE
RS • RETRY SUCCESSFUL
S • SIGN
STE • SEGMENT TABLE ENTRY
STN • SEGMENT TABLE NUMBER
TAG • DESCRIPTOR TYPE
UM • UNDERFLOW MASK

(AS ORIGINALLY FILED)

570331

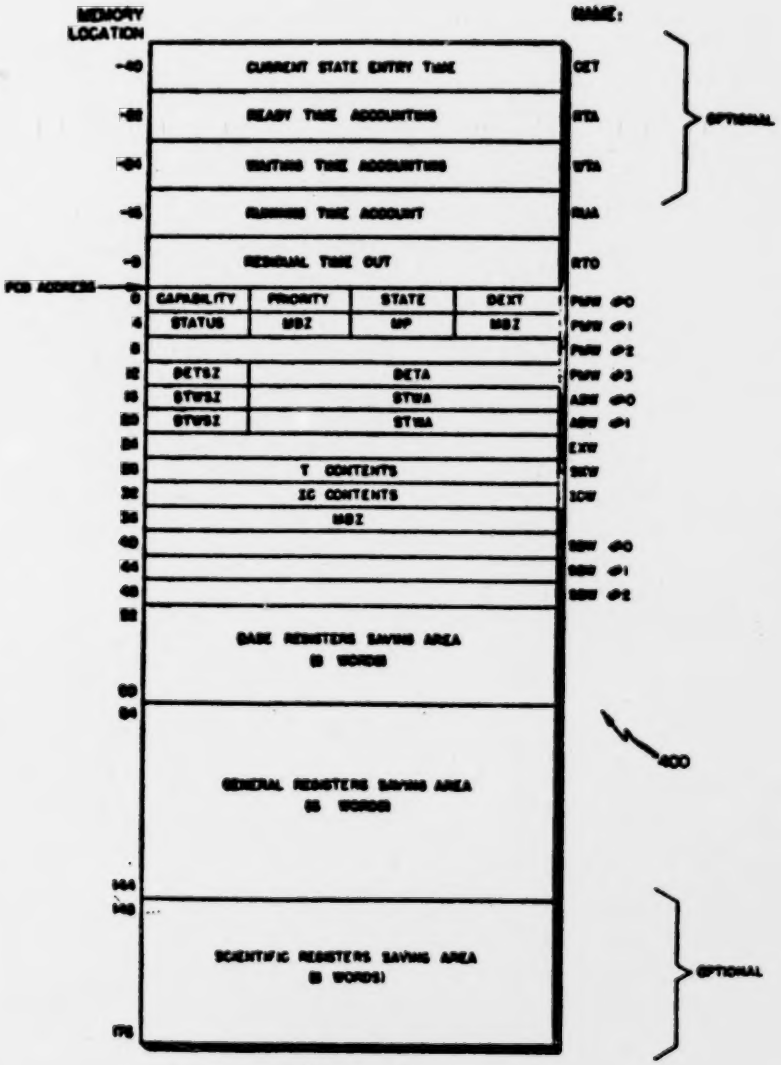


FIG. 4

570371

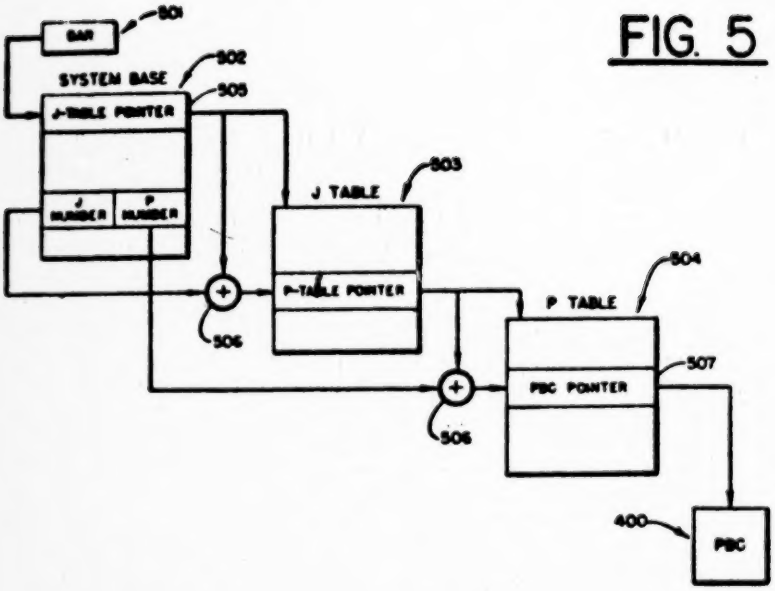


FIG. 5

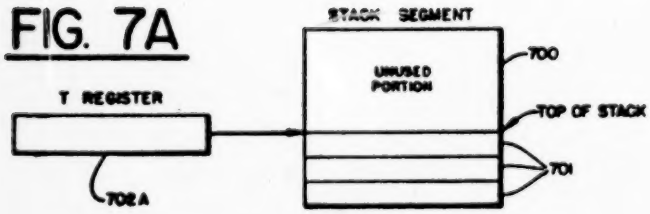


FIG. 7A

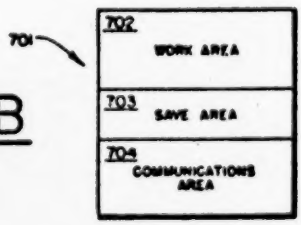
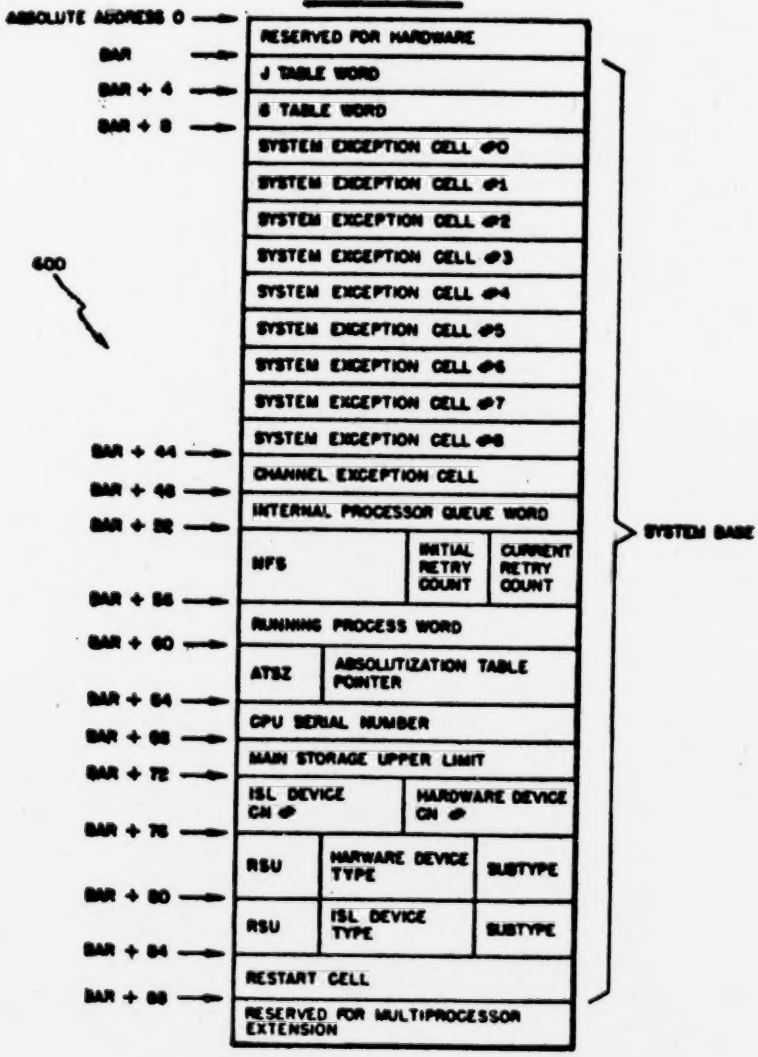


FIG. 7B

(48 ORIGINALLY FILED)

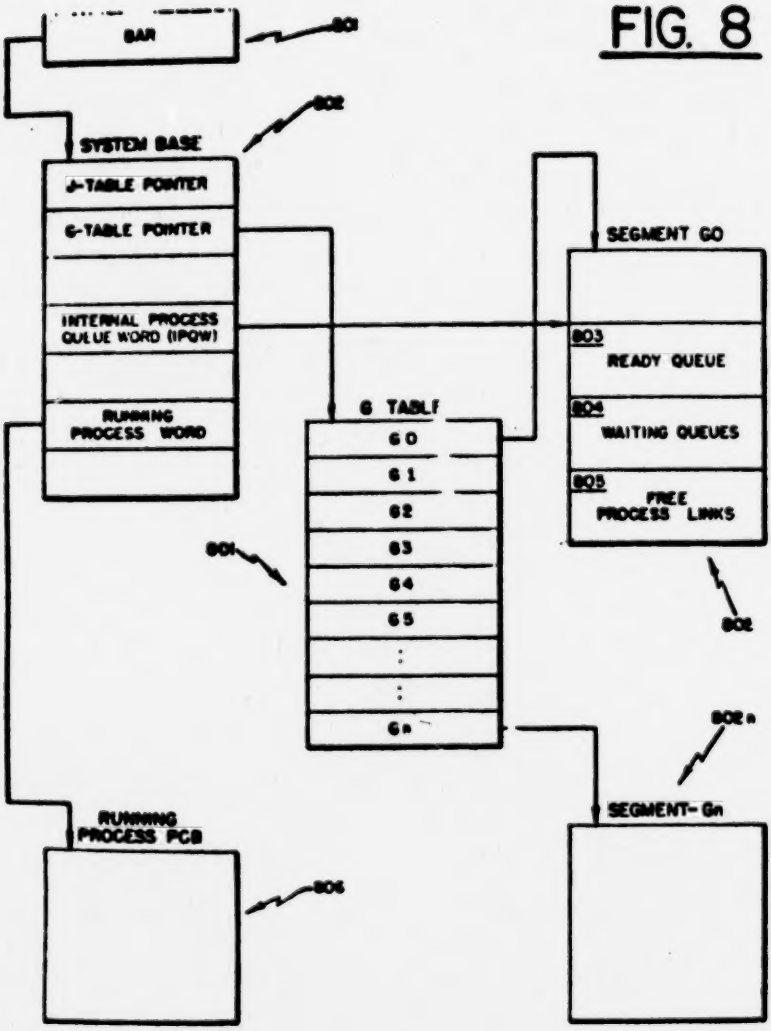
FIG. 6



AS ORIGINALLY FILED

57C331

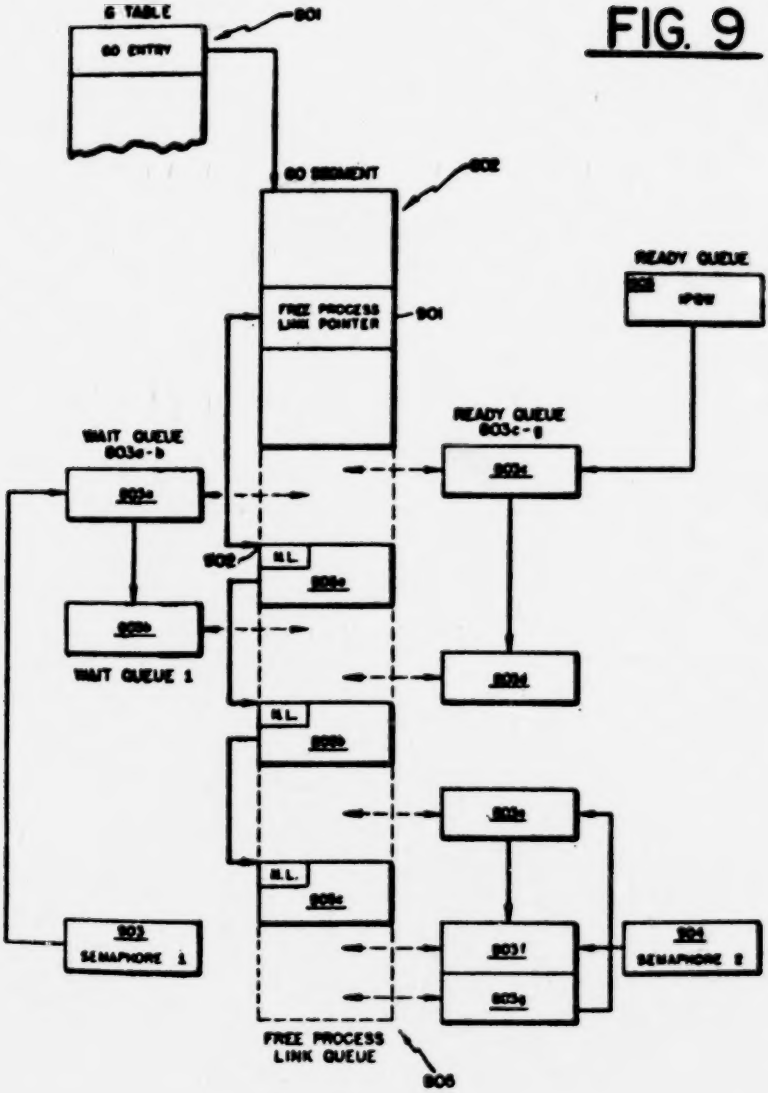
FIG. 8



PRINT OF DRAWING AS ORIGINALLY FILED

570378

FIG. 9



AS ORIGINALLY FILED

57C371

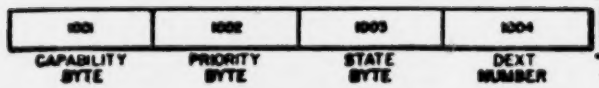


FIG. 10a

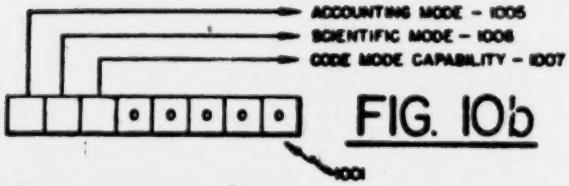


FIG. 10b

FIG. 10c

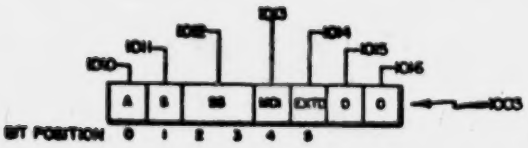


FIG. 10d

FIG. 10e

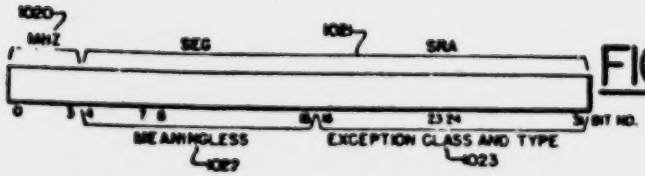


FIG. 10f

AS ORIGINALLY

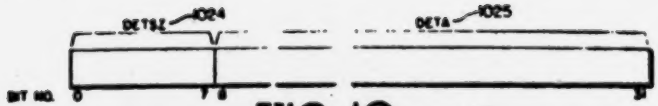


FIG. 10g

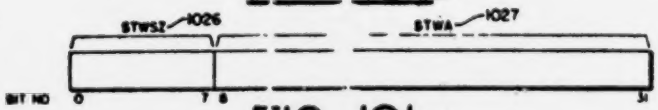


FIG. 10h

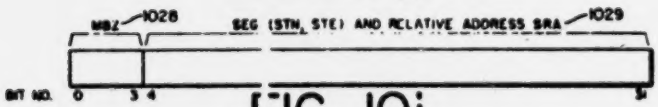


FIG. 10i

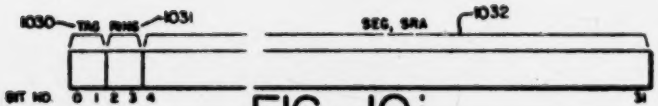


FIG. 10j

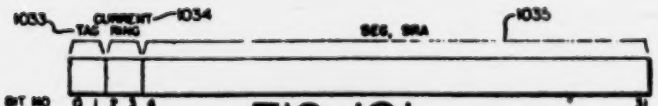


FIG. 10k

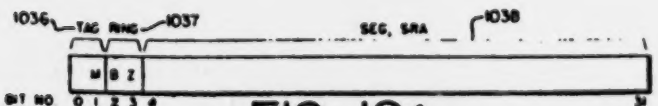


FIG. 10l

AS ORIGINALLY FILED

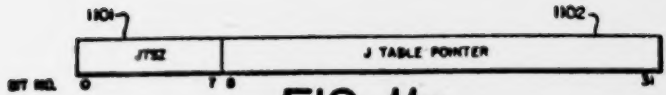


FIG. 11a

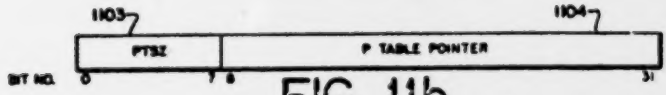


FIG. 11b

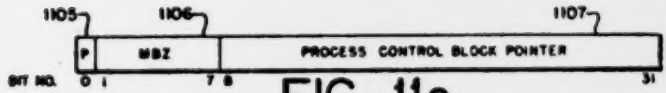


FIG. 11c

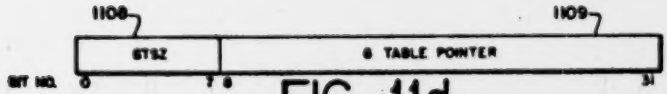


FIG. 11d

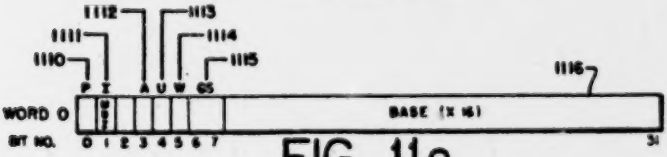


FIG. 11e

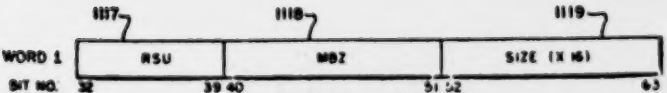


FIG. 11f

COPY OF DRAWING
AS ORIGINALLY FILED

570371

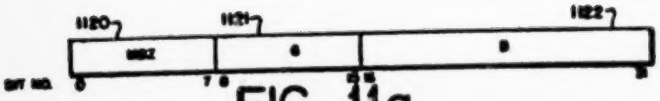


FIG. 11g

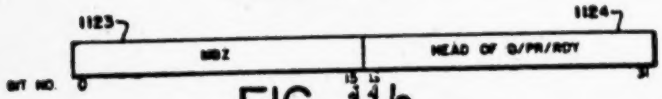


FIG. 11h

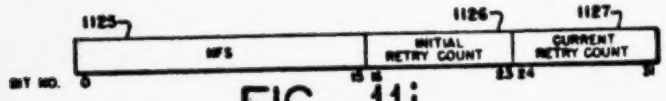


FIG. 11i

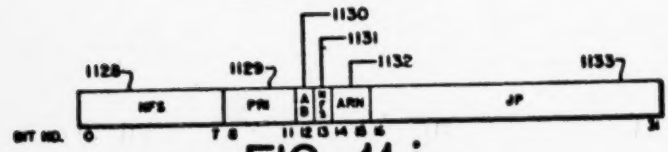


FIG. 11j

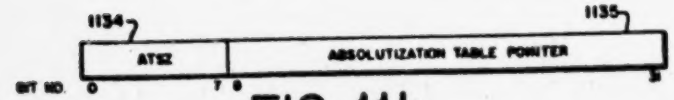


FIG. 11k

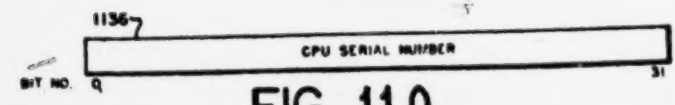


FIG. 11ℓ

PRINT OF DRAWING
NOT ORIGINALLY FILED

57:331

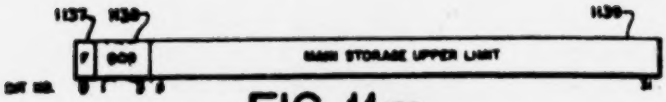


FIG. 11m

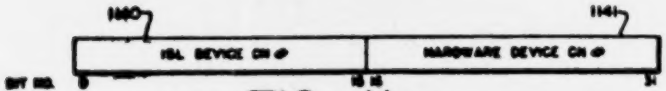


FIG. 11n

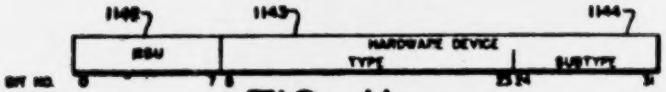


FIG. 11o

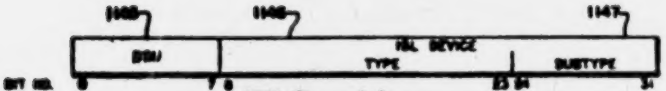


FIG. 11p

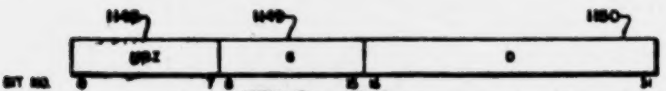


FIG. 11q

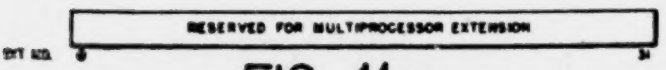
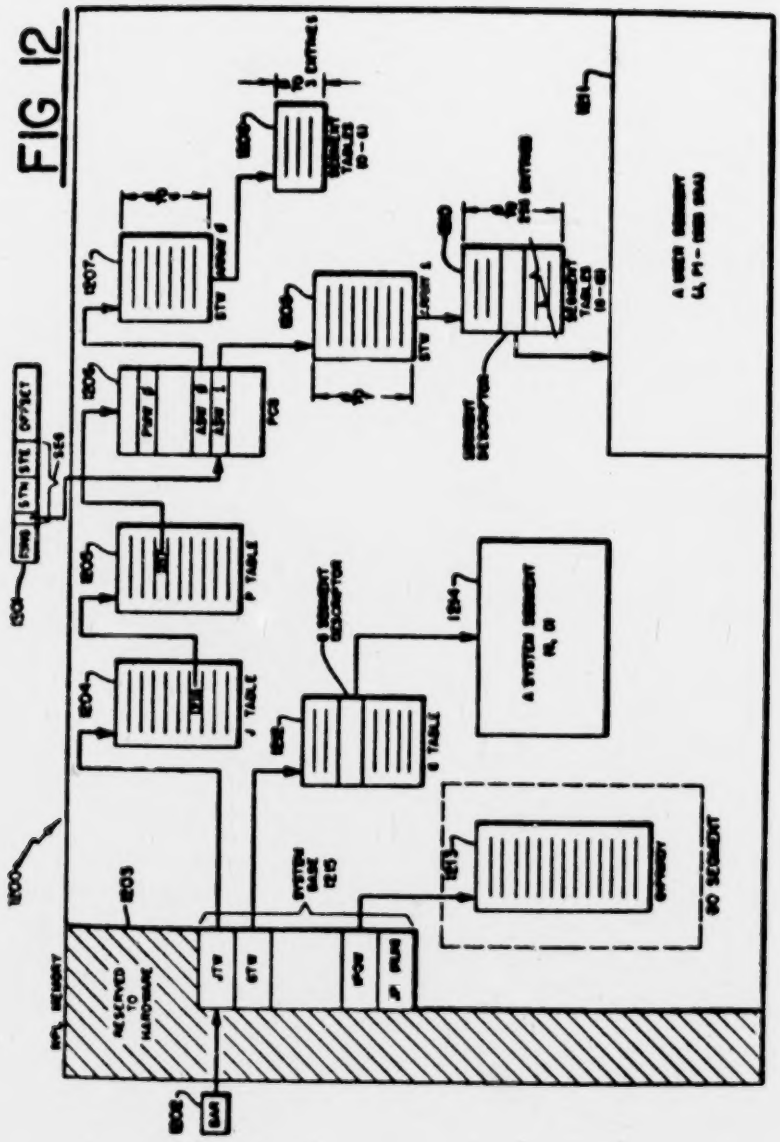


FIG. 11r

PATENT DRAWING
AS ORIGINALLY FILED

57C3'11

FIG 12



POINT OF DRAWING AS ORIGINALLY FILED

570371

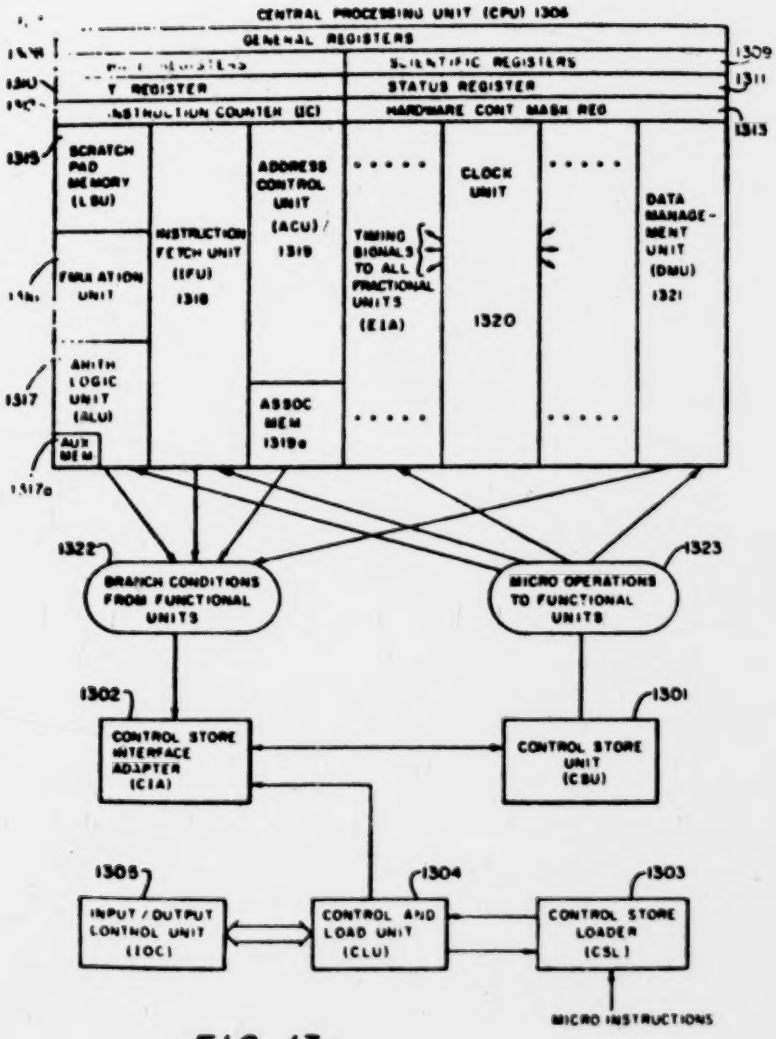


FIG. 13a

FRONT OF DRAWING
AS ORIGINALLY FILED

07C331

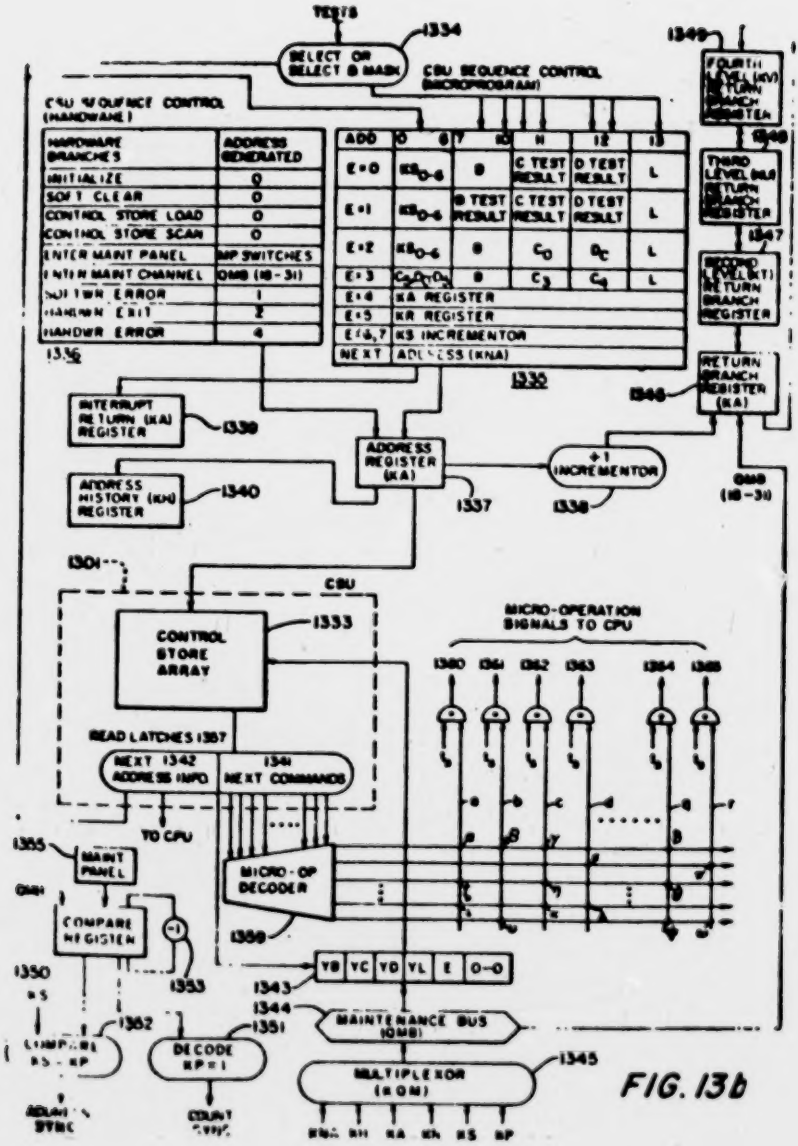


FIG. 13b

570331

PRINT OF DRAWING
WAS ORIGINALLY FILED

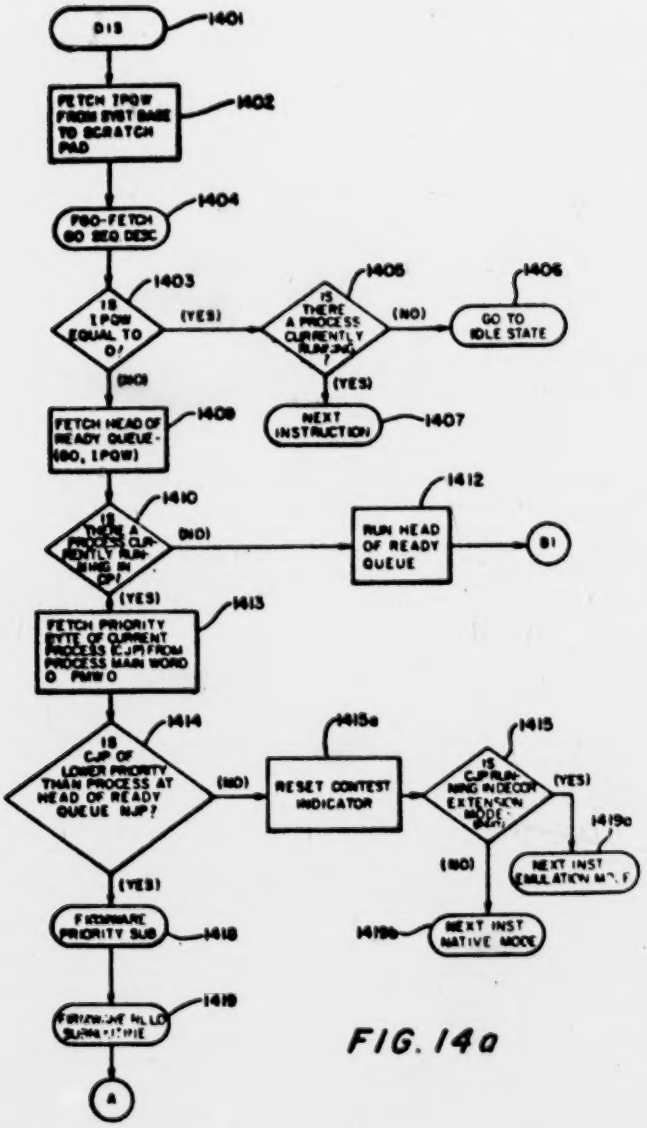


FIG. 14a

PRINT OF DRAWING
WAS ORIGINALLY FILED

57:331

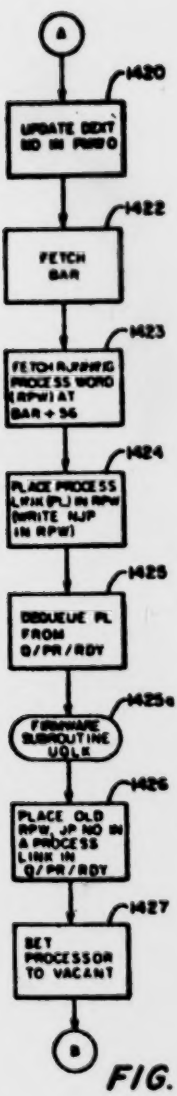


FIG. 14b

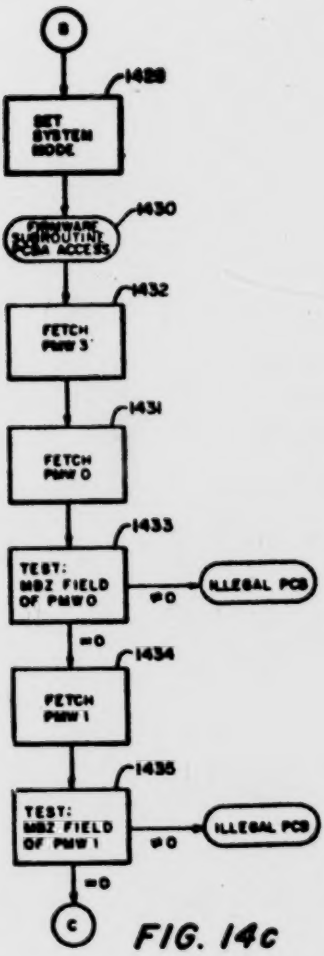


FIG. 14c

PRINT OF DRAWING
AS ORIGINALLY FILED

57C371

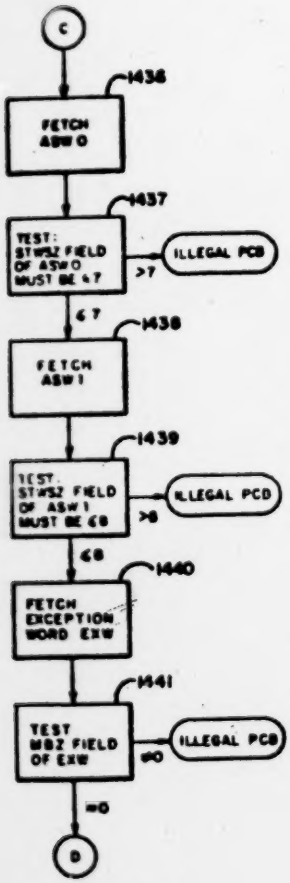


FIG. 14d

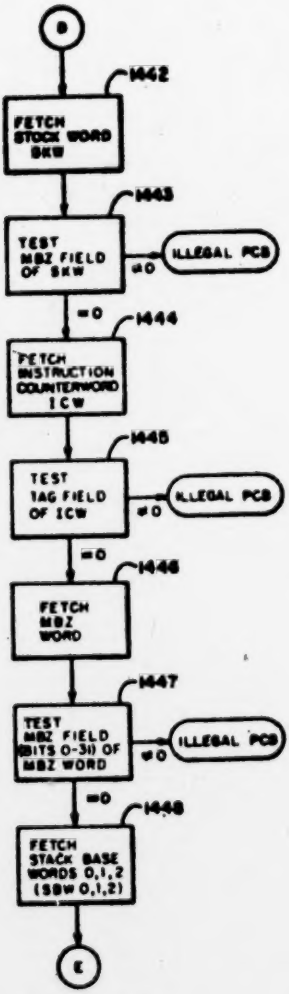


FIG. 14e

PATENT OF DRAWING
WAS ORIGINALLY FILED
570,131

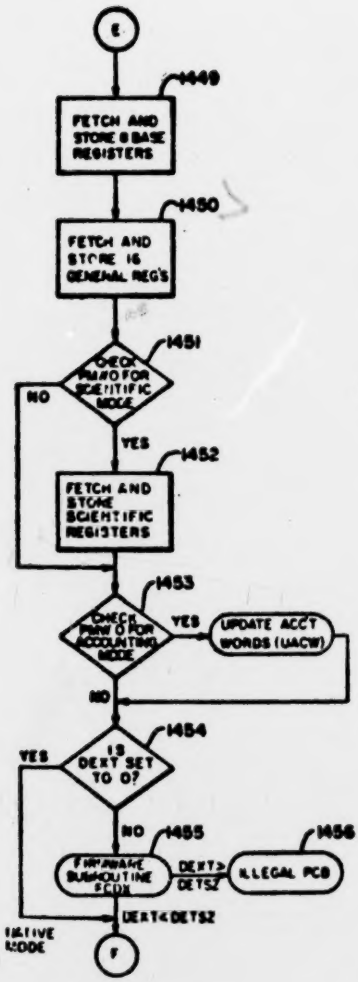


FIG. 14f

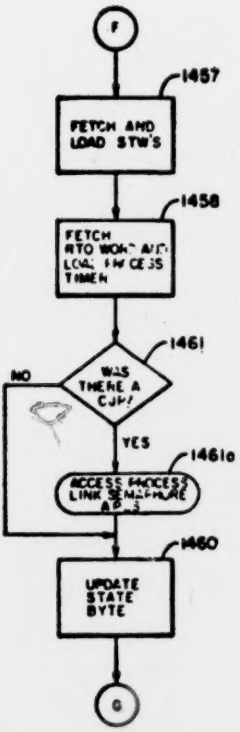


FIG. 14g

WAS ORIGINALLY FILED

570371

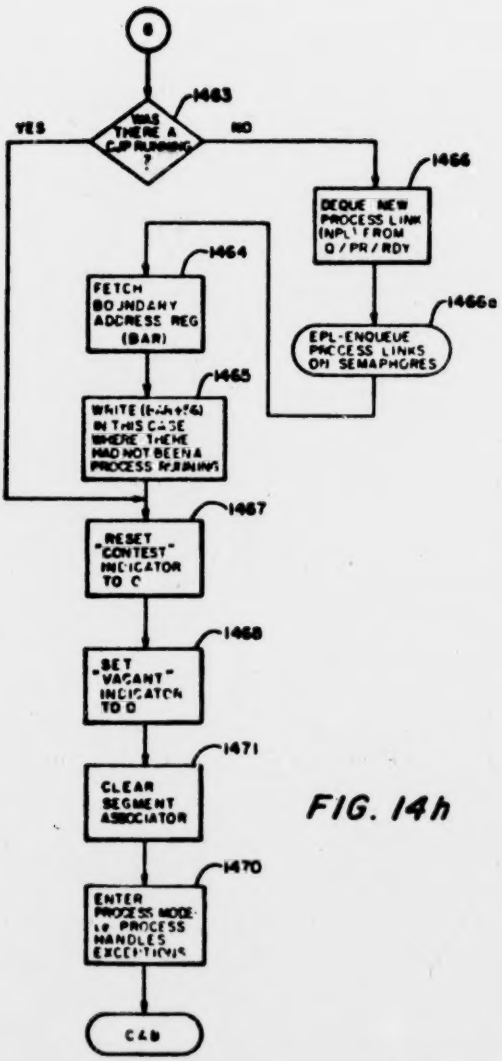


FIG. 14h

PATENT OF INVENTION
AS ORIGINALLY FILED

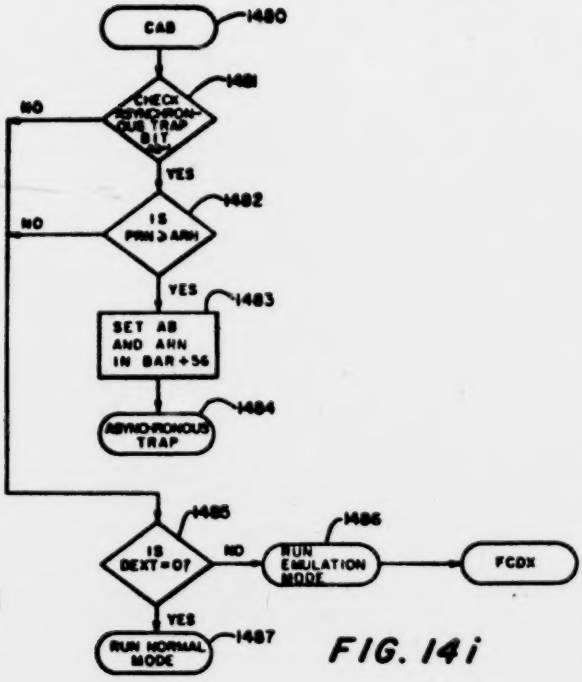


FIG. 14i

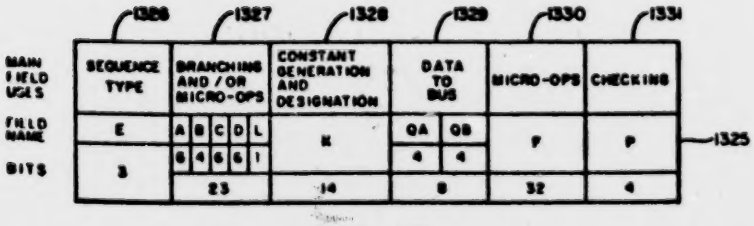


FIG. 13c

PRINT OF DRAWING AS ORIGINALLY FILED

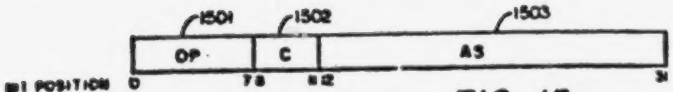


FIG. 15a

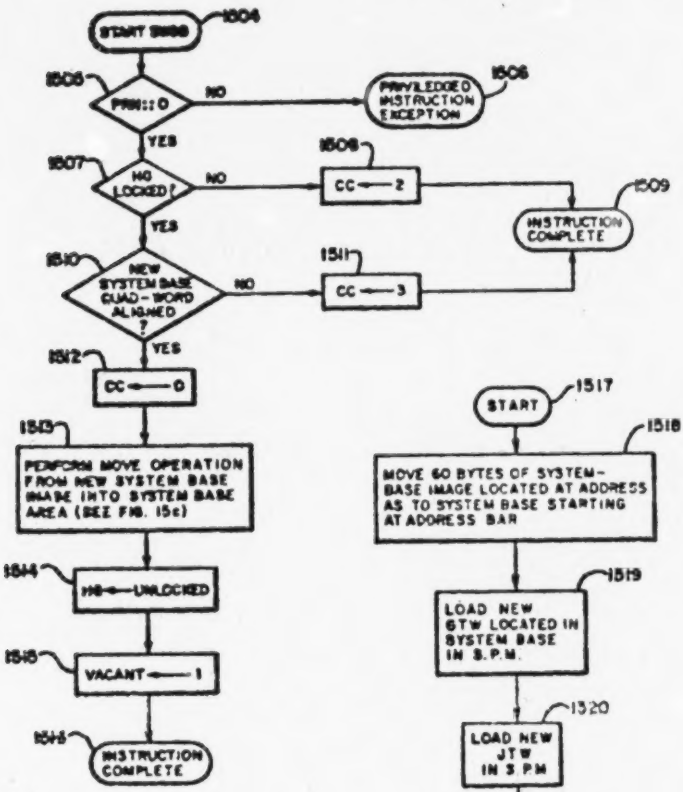


FIG. 15b

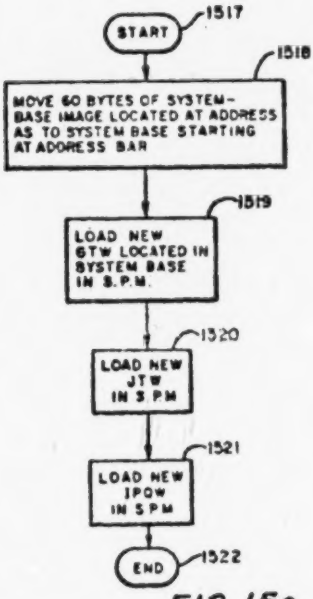


FIG. 15c

LETTER OF EXAMINER, MARCH 12, 1976.

☒ This application has been examined

Responsive to communication filed [illegible] ☐ this action is made final.

A SHORTENED STATUTORY PERIOD FOR RESPONSE TO THIS ACTION IS SET TO EXPIRE

THREE (3) MONTH(S) ~~XXXX~~ FROM THE DATE OF THIS LETTER.

PART I

The following attachments(s) are part of this action:

a. ☒ Notice of References Cited, Form PTO 892.

b. ☐ Notice of Informal Patent Drawing, PTO 948

c. ☐ Notice of Informal Patent Application, Form PTO 152.

d. ☐

PART II

Summary of Action

1. ☒ Claims 1 - 6 are presented for examination.

2. ☐ Claims _____ are allowed.

3. ☐ Claims _____ would be allowable if amended as indicated.

4. ☒ Claims 1 - 6 are rejected.

5. ☐ Claims _____ are objected to.

6. ☐ Claims _____ are subject to restriction or election requirement.

7. ☐ Claims _____ are withdrawn from consideration.

8. ☐ Since this application appears to be in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under Ex parte Quayle, 1935 C.D. 11, 453 OG. 213.

9. ☐ Since it appears that a discussion with applicant's representative may result in agreements whereby the application may be placed in condition for allowance, the examiner will telephone the representative within about 2 weeks from the date of this letter.

10. ☐ Receipt is acknowledged of papers under 35 USC 119, which papers have been placed of record in the file.

11. ☐ Applicant's claim for priority based on an application filed in _____ on _____ is acknowledged. It is noted, however, that a certified copy as required by 35 USC 119 has not been received.

12. ☐ Other

1) Due to the length of the specification it has not been checked to the extent of determining all minor errors. Applicants are requested to review the entire specification and correct any errors noted.

2) The Abstract of the Disclosure should be reviewed and corrected to more specifically and clearly state the character of the subject matter deemed to be new in the art.

3) In the specification, pages 2-4 and various others throughout the disclosure, application serial numbers and filing dates have been omitted.

In lines 1 and 2, page 2, the purpose of the procedure to allow incorporation by reference to previous patents and applications was to reduce the length of disclosure in related applications and simplify the examining and printing procedures involved. Applicants have incorporated by reference numerous applications and/or patents which do not appear to be "essential subject matter" relative to the disclosure of the instant invention. If these prior applications are to be referenced as related applications then it should be so indicated rather than incorporating all of them by reference.

4) The specification is objected to under 35 USC 112 as failing to set forth the subject matter of the invention in such terms as to enable one to make and use the invention without undue experimentation. The basic elements of the system are disclosed in no more detail than hollow rectangles and the complex interrelationship between the various elements is disclosed only in functional statements of desired results. For example, the control unit 130 of Figure 1 is stated to contain firmware which initializes the system, controls the CPU 104 and IOC 105, and encodes an instruction set. However, the elements of the control unit and the instructions for configuring the various elements have not been disclosed in sufficient detail to enable one of ordinary skill in the art to reconstruct the invention.

Applicants have referred to basic data structures on page 11 and following disclosure as being data formats,

software visible registers, and instruction formats. This use of the term "structures" to refer to data formats, etc. is carried over into the claims and results in confusion as to the nature of the invention. The subject matter disclosed in Figure 2 must be deemed in the nature of data format and not of the nature of apparatus as usually intended by use of the word "structure." Clarification is requested.

5(Claims 1-6 are rejected as drawn to subject matter which is inadequately disclosed, 35 USC 112, first paragraph.

6) Claims 1-6 are further rejected under 35 USC 112, second paragraph, as failing to particularly point out and distinctly claim the subject matter sought to be patented.

The structural connections and cooperative relationship between the various elements comprising the claimed combination have not been clearly established. Although the claims are recited as a combination of devices in the means plus statement of function language, most of the claim language does not recite "structure" in the sense of apparatus but rather uses "means" to refer to software or firmware instructions. Applicant is therefore requested to point out the structure in the drawings upon which the various "means" are to be read.

7) Claims 1-6 are further rejected under 35 USC 101 as drawn to non-statutory subject matter.

The Supreme Court in *Gottschalk* v. *Benson*, 175 USPQ 673, stated that Court's views on the patentability of certain claims which set forth an invention related to the processing of data by a programmed computer.

The Court in Benson addressed itself to the broad question of patentability of computer programs and stated that if such subject matter is to be patented, considerable programs are raised which only committees of Congress can manage. Justice Douglas, spe king for the Court, declined to decide such policy matters and refused to extend the patent statutes to embrace computer programming, absent considered action by the Congress.

8) Claims 1-6 are further rejected under 35 USC 102 as unpatentable over the Hauck patent. The apparatus in Figure 1 of Hauck cooperates to perform the function of applicant's claimed invention.

9) The additional art cited, although not applied against the claims, is considered pertinent in showing related systems.

10) All claims have been rejected. This application should be put in condition for final disposition.

/s/ Charles E. Atkinson
CHARLES E. ATKINSON
EXAMINER
GROUP ART UNIT 236

NOTICE OF REFERENCES CITED.

U.S. PATENTS

*		PATENT NO.	DATE	PATENTEE	CLASS	SUB-CLASS	FILING DATE IF APPROPRIATE
	A	3,651,473	3/1972	Faber	340	172.5	
	B	3,665,487	5/1972	Campbell et al.	340	172.5	
	C	3,670,307	6/1972	Arnold et al.	340	172.5	
	D	3,735,364	5/1973	Hatta et al.	340	172.5	
	E	3,725,876	4/1973	Hauck	340	172.5	
	F	3,786,427	1/1974	Schmidt et al.	340	172.5	
	G	3,786,813	1/1974	Cole et al.	340	172.5	
	H	3,889,243	6/1975	Drimak			
	I						
	J						
	K						

FOREIGN PATENTS OR PUBLISHED APPLICATIONS

	PATENT NO.	DATE	COUNTRY	NAME	CLASS	SUB-CLASS	PERTINENT SHTS. DWG	PERTINENT PP. SPEC.
L								
M								
N								
O								
P								
Q								

* OTHER REFERENCES (Including Author, Title, Date, Pertinent Pages, Etc.)

R

S

T

U

EXAMINER	DATE	
C.E. Atkinson	12/4/75	☐ CHECK HERE IF FOREIGN EXCHANGE

* ATTENTION R.O.B.: Do not furnish copy of this box is checked.

AMENDMENT, AUGUST 11, 1976.

August 10, 1976

Commissioner of Patents & Trademarks
Washington
District of Columbia 20231

Dear Sir:

Responsive to the Office Action of 3/12/76 the due date of which was extended two months from 6/12/76 to and including 8/12/76, please amend the above-identified application as follows:

IN THE SPECIFICATION

Page 1, line 2, delete line "A model-idenpendent instruction in firmware" and substitute therefore—Firmware-

Page 1, line 6, start a new paragraph as follows:

Hardware structures centralled by a microprogrammed central store, which provides for the generation of signals which cause the creation of an image in main memory of a communication center comprised of registers at another location. The microprogrammed control store further generates signals which causes the interchange of the information between the communication center and its image.

Page 2, line 1, delete "incorporated by reference" and substitute therefor—related—.

Page 2, line 23, after "invented by" insert—Pravinsinh L. Parmar, et al—.

Page 2, line 24, delete "the same date as the instant invention" and substitute therefor—December 12, 1973—.

Page 2, line 25, before "and assigned" insert 424,239 and issued into Patent No. 3,916,385 on 10/28/75,.

Page 2, line 26, after "invented by" insert—Jacques Bienvenu—.

Page 2, line 27, after "filed on" insert—5/16/74—.

Page 2, line 26, after "Number" insert—470,496—.

Page 2, line 29, delete "Information in a Multiprogram Multiprocessor Computer System" and substitute therefor Data in an Information Multiprocessing System by Implementing a Concept of Rings to Represent the Different Levels of Privileges Among Processes.

Page 2, line 30, after "invented by" insert—Marc Appell—.

Page 2, line 30, after "filed on" insert—December 2, 1974—.

Page 2, line 31, after "Serial No." insert—528,953—.

Page 3, delete lines 1-3 as follows—"Ring Crossing Mechanism" invented by ————, filed on ———— and having Serial No. ———— and assigned to the same assignee named herein.—.

Page 3, line 4, delete "Operations" and substitute therefor—Mechanism—.

Page 3, line 5, before "filed on" insert—Marc Appell—.

Page 3, line 8, after "filed on" insert—December 3, 1974—.

Page 3, line 8, after "Number" insert—529,256—.

Page 3, line 10, after "Resources" insert—To Job Step in a Multiprogram Computer—.

Page 3, line 10, after "invented by" insert—Jean-Pierre Onimus—.

Page 3, line 11, after "filed on" insert—December 2, 1974—.

Page 3, line 11, after "Number" insert—529,255—.

Page 3, delete lines 13-16 as follows—"Data Processing Memory Management System" invented by ———— ————, filed on ———— and having Serial Number ———— and assigned to the same assignee named herein.—.

Page 3, line 17, delete "Computer" and substitute therefor—Data Processing.

Page 3, lines 17-18, delete "Controlled by Reloadable Site-Dependent Information Structures".

Page 3, line 19, before "filed on" insert—William Woods—.

Page 3, line 19, after "filed on" insert—December 2, 1974.

Page 3, line 20, before "and assigned" insert—528,-814—.

Page 3, line 21, after "invented by" insert—Claude Carre—.

Page 3, line 23, before ", and having" insert—December 2, 1974—.

Page 3, line 22, after "Number" insert—528,951—.

Page 3, delete lines 24-27 as follows—"Absolutization During Initialization", invented by ————, filed on ————, and having Serial Number ———— and assigned to the same assignee named herein—.

Page 4, line 4, after "invented by" insert—Claude Carre—.

Page 4, line 5, after "filed on" insert—December 2, 1974—.

Page 4, line 5, after "Number" insert—528,951—.

Page 4, line 11, delete "Absolutization During Initialization" substitute therefor—Arrangement for Initializing a Data Processing System—.

Page 4, line 12, after "invented by" insert—William Woods—.

Page 4, line 12, after "filed on" insert—December 2, 1974—.

Page 4, line 13, after "Number" insert—528,814—.

Page 6, line 2, after "application" insert—references 12 and 14 on page 3—.

Page 12, line 21, after "No." insert—470,496—.

Page 12, line 22, before "entitled" insert—5/16/74—.

Page 17, line 31, after "No." insert—528,953—.

Page 17, line 31, after "Filed" insert—December 2, 1974—.

Page 18, line 1, delete "Information in a Multiprogram, Multiprocessor Computer System" and substitute therefor Data in an Information Multiprocessing System by Implementing the Concept of Rings to Represent the Different Levels of Privilege.

Page 18, lines 2-4, delete—; also see patent application no. ————, filed ————, entitled "Ring Crossing Mechanism" both ————.

Page 19, line 14, after "no." insert—470,496—.

Page 19, line 14, after "filed" insert—5/16/74—.

Page 19, line 15, delete "; also see patent no." and substitute therefor—.—.

Page 42, line 14, after "No." insert—470,496—.

Page 42, line 14, after "filed on" insert—5/16/74—.

Page 54, line 24, after "No." insert—528,814—.

Page 54, line 24, delete "System Initialization Procedure" and substitute therefor—Arrangement for Initializing a Data Processing System—.

Page 54, line 26, delete "Start and Suspend Instruction" and substitute therefor—Process Management Instruction for a Central Processor—.

Page 54, line 29, after "No." insert—529,017 filed on December 2, 1974—.

Page 54, lines 29 and 30 delete "Start and Suspend Instruction" and substitute therefor "Process Management Instruction for a Central Processor" also assigned to the same assignee named herein.

Page 55, line 4, after "No." insert—529,256 filed on 12/2/74—.

Page 55, lines 4 and 5, delete "P and V Operations of Semaphores" and substitute therefor—Process Synchronization Utilizing Semaphores—.

Page 64, line 6, delete "16" and substitute therefor—15—.

Page 79, between lines 5 and 6 insert Appendix I as follows:

APPENDIX I

The following information is provided as background information to better understand the invention, although such information is not necessary to practice the invention.

Further Information with Respect to Figure 1:

(1) Emulators (112)

(a) Emulators are found in the IBM system/370. See page 265 of a book entitled *Computer Organization and the System/370*, by Harry Katzan, Jr., in 1971 by Van-Nostrand Reinhold Company of New York, New York.

(b) Emulators are also described in the following U.S. patents:

3,374,466 issued 3/19/68;
3,543,245 issued 11/24/70;
3,698,007 issued 10/10/72;
3,721,961 issued 3/20/73.

(2) Computation Unit (111)

(a) Computational units are well known in the art and are also known as Arithmetic and Logic Unit (ALU). See the following U.S. patents:

3,400,371 issued 9/3/68;
3,404,378 issued 11/1/68;
3,238,508 issued 3/1/66;
3,201,762 issued 8/17/65.

(3) Control Unit (130)

(a) Typical control units are described in a book entitled *Microprogramming: Principles and Practices*, by Samir S. Husson, published in 1970 by Prentice-Hall Inc., of Englewood Cliffs, New Jersey; see also a book entitled *Computer Organization and Microprogramming*, by Yao-Yaohan Chu published in 1972 by Prentice-Hall Inc., of Englewood Cliffs, New Jersey. Typical state of the art control units are disclosed in the following U.S. patents:

3,400,371 issued 9/3/68;
3,370,274 issued 2/20/68;
3,380,025 issued 4/23/68;
3,268,872 issued 8/23/66;
3,302,183 issued 1/31/67;
3,391,394 issued 7/2/68;
3,302,183 issued 1/31/67;
3,477,063 issued 11/4/69;
3,391,394 issued 7/2/68;
3,646,522 issued 2/29/72.

(4) Buffer Store (110)

(a) See reference 1, page 2, of this application.

(b) See also pages 296-311 of the above referenced book on *Computer Organization and Microprogramming*, by Yao-Yaohan Chu.

(c) Typical buffer stores are also disclosed in the following U.S. patents:

3,411,142 issued 11/12/68;
3,609,665 issued 9/28/71;
3,508,220 issued 4/21/70.

(5) Main Memory Synchronizer (109)
(a) See reference 4, page 2, of this application
(b) See also U.S. patent:

3,449,723 issued 6/10/69.

(6) Associative Memory (132)
(a) Associative memories are well known in the prior art, see pages 273-284 of the above referenced book on *Computer Organization and Microprogramming*, by Yao-Yaohan Chu; see also the following U.S. patents:

3,648,254 issued 3/7/72;
3,456,243 issued 7/15/69;
3,253,265 issued 5/24/66;
3,248,704 issued 4/26/66;
3,241,123 issued 3/15/66;
3,170,142 issued 2/16/65.

(7) Address Control Unit (ACU) 131
(a) See reference 6, page 2, of this application.
(8) Input/Output Control Unit (105)
(a) Typical I/O control units are disclosed in the following U.S. patents:

3,559,187 issued 1/26/61;
3,593,299 issued 7/13/71;
3,251,040 issued 5/10/66;
3,573,741 issued 4/6/71;
3,444,520 issued 5/13/69;
3,680,054 issued 7/25/72;
3,359,543 issued 12/19/67;
3,651,480 issued 3/21/72;
3,248,701 issued 4/26/66;
3,254,330 issued 5/31/66.

(9) Peripheral Control Unit (106)
(a) See Chapter 10 of the above referenced book on *Computer Organization and Microprogramming*, by Yao-Yaohan Chu.

(b) Typical peripheral control units including device adapters are further disclosed in the following U.S. patents:

3,742,457 issued 6/26/73;
3,753,236 issued 7/14/73;
3,408,632 issued 10/29/68;
3,614,740 issued 10/19/71.

(10) Device Adapters (107)

(a) See reference 9, of this page. See also the following U.S. patents:

3,573,741 issued 4/6/71;
3,680,054 issued 7/25/72.

(11) Main Memory (102)

(a) See references 2-7, inclusive on page 2 of this application.

Background Information Relative to Figure 13a

In addition to the references already cited in this Appendix, the following are also included:

(12) General Registers (1307-1311)

(a) Registers are well known in the computer art. Typical ones are disclosed in the following U.S. patents:

3,117,307 issued 1/7/64;
3,108,193 issued 10/22/63;
3,383,521 issued 5/14/68;
3,289,010 issued 11/29/66;
3,258,696 issued 6/28/66.

(13) Hardware Control Mask Register (1313)

(a) See also U.S. patents:

3,351,915 issued 11/7/67;
3,602,898 issued 8/31/71.

(14) Instruction Counter (1312)

(a) Instruction counters are well known in the computer art. See for example U.S. patent:

3,162,841 issued 12/22/64.

(15) Scratch Pad Memory (1315)

(a) Typical Scratch Pad Memories are disclosed in the following U.S. patents:

3,248,709 issued 4/26/66;
3,351,909 issued 11/7/67.

(16) Instruction Fetch Unit (1319)

(a) Instruction fetch units are well known in the computer art. See for example pages 339-347 of the above referenced book on *Microprogramming: Principles and Practices*, by Samir S. Husson.

(b) Typical instruction fetch units are also disclosed in the following U.S. patents:

3,609,700 issued 9/28/71;
3,614,747 issued 10/19/71.

(17) Clock Unit and Timing Circuits (1320)

(a) The computer art is replete with clock and timing circuits. The following U.S. patents disclose typical clock and timing circuits:

3,355,710 issued 11/28/67;
3,417,379 issued 12/17/68;
3,254,329 issued 5/31/66;
2,297,305 issued 3/1/66;
2,781,446 issued 2/12/57;
3,413,454 issued 11/26/68;
3,555,594 issued 11/28/67.

(18) Data Management Unit (1321)

(a) See the following U.S. patents for typical data management and control systems:

3,618,045 issued 11/2/71;
3,665,487 issued 5/23/72;
3,639,912 issued 2/1/72;
3,495,220 issued 2/10/70;
3,599,158 issued 8/10/71.

(19) Control Store Unit (1301)

(a) Control stores are well known in the computer art. See prior reference books entitled *Microprogramming: Principles and Practices*, by Samir S. Husson and Com-

puter Organization and Microprogramming, by Yao-Yaohan Chu.

(b) The following U.S. patents also disclose typical control stores:

3,380,025 issued 4/23/68;
3,469,247 issued 9/23/69;
3,160,858 issued 12/8/64.

(20) Control Store Interface Adapter (1302)

(a) See the following U.S. patents:

3,636,524 issued 1/18/72;
3,353,162 issued 11/14/67;
3,868,649 issued 2/25/75;
3,611,311 issued 11/5/71.

(21) Control Store Loader and Control and Load Unit (1303, 1304)

(a) Program and microprogrammed loaders are well known in the computer art. See for example pages 284-294 of the above-referenced book on *Computer Organization and Microprogramming*, by Yao-Yaohan Chu.

(22) Firmware Computer

(a) The following U.S. patents further disclose microprogrammed computer processors including microprogrammed control units with which the instant invention may be practiced:

3,400,371 issued 9/3/68;
3,656,123 issued 4/11/72.

IN THE CLAIMS

* * * *

For Claims 1, 2 and 4, see Rejected Claims 1, 2 and 4.

REMARKS

Claims 1-6 are still active in the application.

Claims 1, 2 and 4 have been amended.

Reconsideration of the instant application is respectfully requested.

The Applicants have made minor amendments to the specification to correct obvious errors and to complete the information on related applications. Claims 1, 2 and 4 have also been amended for overcoming the Examiner's objection under 35 USC 112. Moreover, Appendix I has been supplied to the specification by amendment. Appendix I supplies the prior art background information which the Examiner has labeled as "hollow rectangles" . . . and "not disclosed in sufficient detail to enable one of ordinary skill in the art to reconstruct the invention." It should be noted that all of this information is background information and pertains mainly to Figures 1, 13a and 13b wherein prior art devices or their equivalents are identified or correspond to the blocks in those figures having the same reference numeral in the Appendix as in the drawing are provided. Since this matter is totally background information and is not necessary to the practice of the invention by a person of ordinary skill in the art without undue experimentation, it entails no new matter with respect to the invention. Accordingly, it is submitted that the Applicants are entitled to have it included with the original specification.

The following arguments are further submitted for the Examiner's consideration relative to the 35 USC 112 insufficiency disclosure. The Applicants have followed 35 USC 112 and established case law which interprets the requirements of 35 USC 112 in making their disclosure. Several cases have spoken with respect to the disclosure. However, before discussing any of these cases, it should be pointed out that the Board of Appeals of the Patent Office has also spoken with respect to the disclosure of complex system inventions In Ex parte Billottet decided 1/26/76 by the Board of Appeals, the Board recognized functional-type block diagrams as a means for disclosing a complex system invention even though the specification does not particularly identify each element represented by the blocks and does not specify particular apparatus intended to carry out each function. The use of block diagrams to disclose the invention is sufficient to comply with the best mode requirements of 35 USC 112, first paragraph, and the requirements of Section

112, second paragraph, to distinctly point out the invention if the individual components of the system are conventional and the invention resides in the particular arrangement of these components.

The Board said, "Block diagrams may be acceptable and, in fact, preferable in cases of this nature where they serve in conjunction with the rest of the specification to enable one skilled in the art to practice the claimed invention. . . drawings and descriptions of circuits that are conventional would in a complex system disclosure such as is here involved, tend to obscure the claimed invention. Here we feel the practice of the claimed invention could be achieved without the necessity of further experimentation itself of an inventive nature."

In In Re Ghiron and Ulrich, 169 USPQ 723, the Court said "Functional-type block diagrams are preferable disclosure if they serve in conjunction with the rest of the specification to enable persons skilled in the art to select prior art apparatus that will produce the results required to practice the claimed process and to practice claimed invention with only a reasonable degree of routine experimentation; with respect to the best mode, the Court said in In Re Bosie, 149 USPQ 789 "Inventor complies with best mode requirement of 35 USC 112 if he does not conceal what he feels is a preferred embodiment of the invention; there is no statutory basis for reading into best mode portion of the statute, a requirement that the mode disclosed be in fact the optimum mode of carrying out invention; *it is error to require that the best mode set forth details not relating to the essence of invention*; there is no basis for best mode rejection merely because the Applicant does not pin-point within the specification some portions that are termed best mode."

Additionally, the Examiner's attention is drawn to U.S. Patent No. 3,891,974 issued June 24, 1975 and assigned to Honeywell Information Systems Inc. In this patent, Figures 1-14h are identical. Moreover, the specification describing these figures is substantially the same. Accordingly, it is submitted that the Patent Office has already placed the stamp of approval on the disclosure pertaining to Figures 1-14H among which are the blocks

the Examiner is questioning. There is therefore, precedence in the Patent Office itself which has recognized as sufficiently disclosed at least those items that the Examiner is questioning in the instant Office Action. For example, the Examiner questioned "the control unit 130 of Figure 1 (which) is stated to contain firmware which initializes the system, controls the CPU 104 and IOC 105, and includes an instruction set. However, the elements of the control unit and the instructions for configuring the various elements have not been disclosed in sufficient detail to enable one of ordinary skill in the art to reconstruct the invention."

The Applicants submit that before it can be established that Applicants have failed to meet the best mode and enabling requirements of 35 USC 112, it must be first determined that which is the subject matter of the present invention as defined by Claims 1-6 (see In Re Bosy, 149 USPQ 792).

Claim 1 specifically claims a data structure in main memory. Webster's Seventh New Collegiate Dictionary, published by G. & C. Merriam Co., of Springfield, Massachusetts define a structure as:

(a) The elements of an entity or the position of such elements in their relationships to each other;

(b) The arrangements of particles or parts in a substance or body;

(c) Something made up of interdependent parts in a definite pattern of organization;

(d) Interrelation of parts as dominated by the general character of the whole.

The data structure system claimed is comprised of the first means which is the structure and arrangement of the running process word register at BAR+56 of the system base structure of Figure 6. The system base structure is described with the particularity of 35 USC 112 beginning on page 35 through 41. Its use is further described beginning on page 41. The running process word register structure is further shown with particu-

larity on Figure 11j and is described with particularity beginning on page 39. The second means which is an element within the first means is part of the running process word register and is identified with reference numeral 1129. The third means within the data structure is the J table word and the G table word at BAR and BAR+4 on Figure 6 and further shown on Figures 11a and 11b and described in the specification. Finally, the fourth means claimed is shown on Figures 13a and 13b is a central store as microprogrammed by the microprogram as disclosed by the flow charts of Figures 15b and 15c. Details of the control store which is microprogrammed is shown on Figure 13b (it should be understood that control stores are old in the art and persons of ordinary skill in the art can microprogram them when provided with the flow charts of the microprogram and this results in the firmware. It should be further noted that firmware should not be equated with the microprogram per se. The Computer Dictionary and Handbook, by Sippl et al. defines firmware as "logic circuits in read-only memory . . . . ."). Figures 13a and 13b are described with great particularity beginning on page 43 through page 53. Moreover, beginning on page 55 an example is taken by applying the microprogram flow chart of Figure 14a and actually microprogramming the control store of Figure 13b. Referring to Figure 14a and to page 55 of the specification the first block 1402 of Figure 14a instructs the microprogrammer to fetch the IPQW word from the system base into scratch pad memory. Beginning on line 16 of page 55 of the specification, there is shown the generation of the micro-operation signal 1360, 1361 to cause the retrieval of the IPQW word from the system base and storage subsystem 102, and transfer it to scratch pad memory 1315. Continuing further in the description of 14a there is similar descriptions with similar particularity to teach a person of ordinary skill in the art how to microprogram the control store or ROM in order to generate the firmware, first of the dispatch unit and more particularly the firmware of the instant invention. In In Re Chilowsky, 108 USPQ 321, the Court said "disclosure of application em-

braces not only what is expressly set forth in words or drawings, but what would be understood by a person skilled in the art." In In Re Karnofsky, 156 USPQ 682 the Court said in part "where one of ordinary skill in the art would know how to select operating conditions so as to achieve a particular result, the failure to include a recitation of some specific operating conditions and specifications cannot give rise to rejection either under 'enabling' or under 'best mode' requirement of 35 USC 112." Morevoer, In Re Karnofsky the Court said "Mere conclusory statements as to level of skill in the art are insufficient to controvert facts presented by Applicant as to sufficiency of specification's disclosure; it is not demonstrated how specification fails to meet the requirements of 35 USC 112." It is respectfully submitted that when the invention is reviewed in a factual manner as herein demonstrated by the Applicants that the specification discloses that which the Applicants consider as their invention with great particularity. A similar analysis may be conducted for the remaining claims.

The Examiner has further objected to the use of the term "basic data structure" as used on page 11 of the specification. Referring to page 11 the title under II is Basic Machine Structures. Continuing under the title the Applicants state "There are typically 3 basic data structures utilized in this hardware: data formats, software visible registers, and the instruction format." Webster's Seventh New Collegiate Dictionary previously referred to gives 2 definitions for "format" as a noun, are of which is as follows: "general plan of organization or arrangement." This is the definition intended by the Applicants and as such taken in context with the specification as a whole no confusion should result. In In Re Chilowsky, 108 USPQ 321 the Court said in part "disclosure of application embraces not only what is expressly set forth in words or drawings, but what would be understood by persons skilled in the art." In In Re Storrs, 114 USPQ 293 the Court in part "certainly required of disclosure by first paragraph of 35 USC 112 is not greater than that which is reasonable, having due regard to subject matter involved; in determining cer-

tainty required it cannot be forgotten that disclosure is not addressed to public generally, but to those skilled in the art." Along on the same subject matter in In Re Honn and Simms, 150 USPQ 652 the Court said in part "Applicants need not expressly set forth in specification that which would be understood by persons skilled in the art."

It is submitted that the Applicants have disclosed their invention in accordance with 35 USC 112 and the applicable case law. Applicants are further directing the Examiner's attention to other cases pertaining to disclosure of invention with requirements of 35 USC 112 as follows:

35 USC 112 CASES

(a) In Re Naquin (158 USPQ 317) 1968;
(b) In Re Ghiron and Ulrich (169 USPQ 793) 1971;
(c) In Re Brown (177 USPQ 691) 1973;
(d) In Re Knowlton (178 USPQ 486) 1973;
(e) In Re Comstock and Gilmer (178 USPQ 616) 1973;
(f) In Re Doyle (179 USPQ 237) 1973;
(g) In Re Brandstadter, Kiencle and Sykes (179 USPQ 286) 1973;

Moreover, the Examiner's attention is further directed to a book entitled, Microprogramming: Principles and Practices, by Samir S. Husson, published by Prentice-Hall Inc., of Englewood Cliffs, New Jersey in 1970. Moreover, the Examiner's attention is directed to an article entitled, Microprogramming Simplifies Control System Design, by Ron Kenny of Micro Data Corporation, Irvine, California. This article was published in Computer Design, the magazine of Digital Electronics, on February, 1975, Volume 14, No. 2. The address of the magazine is 221 Baker Avenue, Concord, Massachusetts.

In view of the above amendments, the discussion and the above authorities cited by the Applicants, the Examiner is respectfully requested to reconsider and remove his rejection based on an inadequate disclosure under 35 USC 112.

Claims 1-6 have further been rejected as drawn to subject matter which is inadequately disclosed under 35 USC 112 first paragraph. It is submitted that the amendment, discussions and authorities cited above also rebut this rejection and accordingly it is respectfully requested that the Examiner reconsider and remove this rejection.

Claims 1-6 have further been rejected under 35 USC 112, second paragraph, as failing to particularly point and distinctly claim the subject matter sought to be patented. The Examiner concludes in this rejection that "means refers to software of firmware instructions." Applicants are therefore requested to point out the structure in the drawings upon which the various means are to be read. This has been done with respect to the discussion supra. For example, Claim 1 as now amended was read in conjunction with the figures in the text of the specification. The first, second and third means are all hardware structure which store coded signals for performing a particular function as sanctioned under 35 USC 112 last paragraph. Software does not store signals. Moreover, the fourth means as now amended is also read upon the microprogrammed control store which generates signals which cause the particular function recited. Certainly this means is not claiming software. However, the fourth means is very definitely claiming firmware. It is not, however, claiming "firmware instructions" as the Examiner alleges. It has shown previously by quoting the Computer Dictionary and Handbook, by Sippl, et al. that firmware is "logic circuits in read-only memory . . ." It was previously shown how any control store having a ROM and more particularly the control store as disclosed in Figure 13a and 13b is microprogrammed with the flow charts of Figures 15b and 15c which then resulted in the first firmware claimed by the fourth means. Note the bare control store or ROM is not claimed and it is admitted is old in the art. Moreover, the flow charts of Figures 15b and 15c are not claimed per se. It is a combination of the microprogrammed control store which when microprogrammed in accordance with the microprogram depicted by flow charts of 15b and 15c results

in the firmware of the fourth means. The Examiner is further referred to the previously referenced book of Husson on "Microprogramming: Principles and Practices" which shows how persons of ordinary skill in the art talk to one another via flow charts in microprogramming control stores. It is submitted therefore that Claims 1-6 recite structure in the sense of apparatus, as clearly demonstrated herein. If the Examiner continues to maintain that this is not the case, he is respectfully requested to point out with particularity wherein the structural connections and cooperative relationships between various elements comprising the claimed combination have not been clearly established. In In Re Warner and Warner, 154 USPQ 174 the Court said "The Patent Office has initial duty of supplying *factual basis* for rejection; it may not, because it may doubt that invention is patentable, resort to speculation, unfounded assumptions or hindsight reconstruction to supply deficiencies in its factual basis, to extent that Office rulings are so supported, there is basis for resolving doubts against their correctness; likewise Court may not resolve doubts in favor of Office determination when there are deficiencies in record as to necessary factual bases supporting its legal conclusion or obviousness." (Underlining added.) Accordingly, in view of the above amendments to the claims, the arguments hereinbefore presented and the legal authorities cited, it is respectfully requested that the Examiner reconsider and remove his rejection under 35 U.S.C. 112 second paragraph.

The Examiner further rejected Claims 1-6 under 35 USC 101 as drawn to non-statutory subject matter. The Examiner said in making this rejection that "The Supreme Court in Gottschalk v. Benson, 175 USPQ 673, stated that Court's view on the patentability of certain claims which set forth an invention related to processing of data by a programmed computer.

"The Court in Benson addressed itself to the broad question of patentability of computer programs and stated that if such matter is to be patented, considerable

questions are raised which only committees of Congress can manage. Justice Douglas, speaking for the Court, declined to decide such policy matters and refused to extend the Patent Statutes to embrace computer programming, absent considered action by the Congress."

The Applicants traverse the Examiner's rejection of Claims 1-6 under 35 USC 101 as being drawn to nonstatutory subject matter. In the first instance it should be noted that the Benson application, upon which the Benson and Tabbot decision was based, included apparatus claims which were not the subject of the decision made by the Supreme Court. Apparently, it was considered that the apparatus claims did, in fact, define patentable subject matter while the process claims did not. However, the Benson invention was directed to a type of computer invention which happened to be implemented by means of a program.

As defined by Applicants' claims, the present invention is directed to hardware structures controlled by a microprogrammed control system which provides for the generation of signals which cause the creation of an image of a communication center at another location and for the generation of signals which causes the interchange of the information between the communication center and its image. (See Claim 2). It is difficult to understand how microprogram control systems can be said to be nonstatutory subject matter or in other words not pertaining to the technological arts. The technology or art to which Applicants' invention is directed is the same technology and art to which several of the patents cited by the Examiner are directed. Applicants submit that the extension of the holding in the Gottschalk v. Benson decision proposed by the Examiner would remove the majority of patents which have issued on computer program inventions (to which this invention does not pertain) from the purview of 35 USC 101.

Applicants submit that the structure and the microprogram control system defined by the Applicants' claims are clearly within the technological arts. The phrase "technological arts" is synonymous with the phrase" useful arts" as it appears in Article 1, Section 8 of the Con-

stitution (see In Re Waldbaum, 173 USPQ 430) and (In Re Johnston, 192 USPQ 272). As the Examiner may have already noted, the decision of Gottschalk v. Benson ruled that the invention as claimed was unpatentable because it was claimed too broadly and, therefore, "would wholly preempt the mathematical formula" and practically "would be a patent of the algorithm" (see In Re Christensen, 178 USPQ 35). Moreover, in its opinion in Dann v. Johnston which appears in Volume 188, No. 13 in a Supplement to the United States Patent Quarterly, Weekly Advance Sheets, dated March 29, 1976, the U.S. Supreme Court stated that in Benson, the respondent sought to patent as a "new and useful process" a method of programming a general purpose computer to convert signals from binary coded decimal form into pure binary form. The Court went on to say, "As we observe, the claims were not limited to any particular art or technology or to any particular apparatus or machinery or to any particular end use as quoted by the Examiner." With that, the Supreme Court finally stated in that regard that "Our limited holding was that the respondent's method was not a patentable 'process' as the term is defined in 35 USC 100(b)." Clearly, Applicants' claims are not concerned with a method of programming a general purpose digital computer to convert signals from binary coded decimal form into pure binary form. The Supreme Court has indicated that the Benson decision should be so limited.

More importantly, it should be noted that the U.S. Supreme Court in the Benson decision went so far as to issue broad exclusionary warnings prior to the final expression of unpatentability. For example, the Court said "it is said that the decision precludes the patent for any programming servicing a computer. We do not so hold." In a similar situation, Justice Frankfurter, in his concurring opinion given in the Supreme Court's decision of Funk Brothers Seed Company v. Kalo Innoculant Co., 76 USPQ 283, cited in the Gottschalk v. Benson decision, warned about the use of such terms as "works of nature" and "Laws of nature." He advised that these are such vague and malleable terms that the arguments drawn from such terms in ascertaining patentability

could fairly be employed to challenge almost every patent. Justice Frankfurter went on to say in that decision that, "Surely the Court does not mean to unwittingly pass on the patentability of such products by formulating criteria by which future issues of patentability may be prejudged.

From the above, it seems that by relying on the Gottschalk v. Benson decision to the extent of concluding that Applicants' claims are not limited to any particular art or technology, in accordance with the guidelines proposed in Gottschalk v. Benson, the Examiner is doing what Justice Frankfurter forewarned. Moreover, the Applicants submit that the Examiner's interpretation of the Benson decision and its application is contrary to what the Supreme Court intended for the reasons stated above.

In addition to the above statements, Applicants assume that by the term "program" the Examiner means its conventional meaning, as for example, one of the definitions set out in the text, in Computer Dictionary and Handbook, by Charles J. Sippl, et al., Copyright 1966 and 1972 by Howard W. Samms & Co., Inc. That is, a program is defined as one of the following definitions:

(1) A plan for the automatic solution of a problem; or,

(2) A set of instructions or steps that tells the computer exactly how to handle a complete problem . . . payroll, production scheduling or other applications; or,

(3) To plan a computation or process from the asking of a question to the delivery of results; or,

(4) To plan the method of attack for a defined problem.

Applicants submit that neither the claims nor specification is directed to a plan or set of instructions or steps, etc. Even, arguendo, if they were (which they are not) it is submitted that programs disclosed in accordance with 35 USC 112 are patentable subject matter under 35 USC 101. Several cases have spoken in this regard; and some of the leading ones are presented below:

CASES RE 35 USC 101

(a) In Re Abrams (89 USPQ 266) 1951;
(b) In Re Shao Wen Yuan (89 USPQ 324) 1951;
(c) In Re Jones (153 USPQ 77) 1967;
(d) In Re Tarczy-Hornoch (158 USPQ 141) 1968;
(e) In Re Prater and Wei (102 USPQ 541) 1969;
(f) In Re Bernhart and Fetter (163 USPQ 611) 1969;
(g) In Re Mahoney (164 USPQ 572) 1970;
(h) In Re Musgrave (167 USPQ 280) 1970;
(i) In Re Foster, et al (169 USPQ 99) 1971;
(j) In Re McIroy (170 USPQ 31) 1971;
(k) In Re Waldbaum (173 USPQ 430) 1972;
(l) In Re Christensen (178 USPQ 35) 1973.

Omitted from the above cases, all of which held that programs were patentable, subject matter is the Benson and Tabbot decision which as previously discussed supra is severely limited to its own fact situation and was decided by the Supreme Court on the basis that the subject matter was a mathematical algorithm which was claimed too broadly so as to preempt all future uses of the mathematical algorithm i.e. the case was a matter of overclaiming.

The holdings of the above cases are still good law and have not been overruled by Benson and Tabbot; as a matter of fact, Benson and Tabbot itself has been severely limited by the In Re Christensen decision and the In Re Johnston decision discussed supra. Apparently, the CCPA has been making law regarding computer programs, and the Supreme Court has seen fit not to disturb this law with the exception of Benson, wherein its decision was really on the ground of overclaiming so as to preempt all uses of the algorithm in a computer and not on the grounds of a pure program per se.

In view of the above discussion wherein it is pointed out that the subject invention is not a mathematical algorithm and is not directed to a program but rather to hardware and firmware and moreover, arguendo, even if it were it still would be patentable subject matter under

35 USC 101. Accordingly, it is respectfully requested that the Examiner reconsider and remove his rejection based on 35 USC 101.

Claims 1-6 have been further rejected under 35 USC 102 as unpatentable over the Hauck patent. The Examiner states "The apparatus in Figure 1 of Hauck cooperates to perform the functions of Applicants' claimed invention." The Examiner's rejection is hereby traversed and very strong exception is taken to the reason given for the Examiner's rejection on this basis. In the first place, the functions of the Hauck device are completely different than the functions of the instant invention. Moreover, the structure and hardware of the Hauck device are completely different than the structure of the instant invention. The Hauck patent discloses a push down stack in main memory wherein are stored "control words" which mark off procedural areas of the stack, operand words for processing and reference words for indicating locations of control words and of operand words. A pair of high speed registers are provided for coupling information to the processor's arithmetic unit for processing. These two registers are linked to the stacks so that they are the top two available locations of the stack. The processor includes a plurality of high speed registers which are linked to the memory stack to form a stack extension. Stored in the stack extension are control words that form a part of an address environment stack history list; parameters that are local to the current addressing environment; and operands that have been called to the top of the stack. Words stored in the stack extension are captured and are not pushed down into the memory stack during the execution of a procedure. Means are provided for addressing each register within the stack extension and for coupling captured information therein to the register which is the top available location of the stack. Means are provided for updating the stack extension when a new procedure is entered so that it will capture the information relating to the new procedure." (See Abstract of the Hauck patent).

Apparently, the Hauck patent discloses an improvement for a stack register by providing a stack extension to the stack register so that a desired item in the stack can be brought to the top of the stack without accessing main memory. See Columns 1-6 of the Hauck patent. An additional feature of the Hauck device is to store into main memory information related to the active stack that will not fit into the stack extension. Hauck states in column 6, lines 33-44 that "The stack extension according to the present invention which captures information, items and prevents them from being pushed down into main memory is important because the first items stored into the stack extension are usually parameters which are most often accessed for use in processing. Hence minimum access time to these parameters is achieved by keeping them in the higher speed stack extension. This is in contrast to the prior art stack extension where the first stored information items are pushed out of the bottom of the stack extension into main memory."

Stated in as few words as possible, it appears to the Applicants that the Hauck device pertains to an improvement of a stack type register wherein an item of information placed in the stack can be brought to the top of the stack or kept there via an extension stack without using indirect addressing techniques and the attendant multiplicity of main memory accesses.

The instant invention is quite different. The instant invention does not use a stack structure. The instant invention provides a communication area communicating between an operating system and a scratch pad memory which is located in the CPU.

This communication system is improved by the combination which causes an image of the communication center to be generated at another location in main memory, wherein the information in the image is then updated and then retransferred back to the communication center and to the scratch pad memory. This is an entirely different and novel operation, having entirely new, different and novel functions, and entirely new, novel and different combination of structures than the device of

Hauck and as a matter of fact any other devices of which the Applicants are aware. The claimed language of the instant invention reflects precisely these novel functions and novel combination of structures. (See Claim 2 in particular). If the Examiner persists in his rejection of Claims 1-6 under 35 USC 102, he is respectfully requested to point out with particularity how Figure 1 of the Hauck patent cooperates to perform the function of Applicant's claimed invention. In view of the amendments made to the claims and in view of the above discussion, it is respectfully requested that the Examiner reconsider and remove his rejection based on 35 USC 102.

References B-H which were not applied against the claims have been carefully reviewed but do not appear to add anything upon which the claims would read or be obvious in view of the reference.

In view of the amendments made herein and the above discussion, it is submitted that all the claims are now in an allowable form and allowance of Claims 1-6 is hereby respectfully requested and that the application be passed to issue at an early date.

If there are any questions which the Examiner feels can be resolved by a telephone call, he is respectfully requested to call the Applicants' Attorney collect at the telephone number listed below.

Respectfully submitted,

/s/ Nicholas Prasinos
Nicholas Prasinos
Registration No. 24,791
Attorney for Applicants

LETTER OF EXAMINER, OCTOBER 27, 1976.

☒ Responsive to communication filed on August 11, 1976.

☒ This action is made final.

A SHORTENED STATUTORY PERIOD FOR RESPONSE TO THIS ACTION IS SET TO EXPIRE THREE (3) MONTH(S) ~~DAYS~~ FROM THE DATE OF THIS LETTER.

FAILURE TO RESPOND WITHIN THE PERIOD FOR RESPONSE WILL CAUSE THE APPLICATION TO BECOME ABANDONED. 35 U.S.C. 133

PART I THE FOLLOWING ATTACHMENT(S) ARE PART OF THIS ACTION:

1. ☐ Notice of References Cited, Form PTO-892.
2. ☐ Notice of Informal Patent Drawing, PTO-948.
3. ☐ Notice of Informal Patent Application, Form PTO-152
4. ☐

PART II SUMMARY OF ACTION

1. ☒ Claims 1 - 6 are pending in the application.

   Of the above, claims _____ are withdrawn from consideration.

2. ☐ Claims _____ have been cancelled.
3. ☐ Claims _____ are allowed.
4. ☒ Claims 1 - 6 are rejected.
5. ☐ Claims _____ are objected to.
6. ☐ Claims _____ are subject to restriction or election requirement.
7. ☐ The formal drawings filed on _____ are acceptable.
8. ☐ The drawing correction request filed on _____ has been ☐ approved. ☐ disapproved.
9. ☐ Acknowledgement is made of the claim for priority under 35 U.S.C. 119. The certified copy has ☐ been received. ☐ not been received. ☐ been filed in parent application; serial no. _____ filed on _____.
10. ☐ Since this application appears to be in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under Ex parte Quayle, 1935 C.D. 11; 453 OG. 213.
11. ☐ Other

1) This is in response to the amendment filed August 11, 1976. Claims 1 through 6 remain in the application.

2) Claims 1 through 6 are again rejected as being directed to non-statutory subject matter under 35 USC 100 and 101.

Applicants contend that "the present invention is directed to hardware structures controlled by a microprogrammed control system which provides for the generation of signals which cause the creation of an image of a communication center at another location and for the generation of signals which causes the interchange of the information between the communication center and its image." Applicants further contend that "the structure and microprogram control system defined by applicants' claims are clearly within the technological arts" and is thus statutory subject matter within the intended bounds of 35 USC 100 and 35 USC 101.

Although applicants have now amended the claims to couch the claimed subject matter in language inferring structural entities and interconnections, the disclosed invention and specification language clearly indicate that the invention resides in a "technique" or "switch base instruction" for changing information in the system base in both the main memory and internal scratchpad registers making the system base in scratchpad memory accessible to software but not visible to software i.e. cannot be read by the software.

In claim 1, the preamble sets for the environment as a multiprogramming computer system comprised of such well known components as a main memory, central processing unit, and scratchpad registers. The subject matter deemed as the invention is then stated as "a data structure." The nature of this subject matter is disclosed beginning on page 35 of the specification and in Figure 6. Beginning in line 12 of page 35, it is stated that "the format of the system base 600 is shown." It is further stated that the "system base resides in absolute main memory and is developed by firmware." The term "data structure" as recited in the claim is comprised of four entities described as "first means", "second means",

"third means", and "fourth means." Whether these so called "means" are hardware structure which store coded signals for performing a particular function or whether they refer to coded signals representing computer control words per se, any life which the claim may have in view of that which is admittedly old in the art, resides in the "technique" (page 5, line 4) or "switch-system base instruction" (page 5, lines 13 and 14) for solving the problem of changing information of the system base which is located in scratchpad registers which software cannot access (page 4, line 20). The invention then is embodied in the coded signals representing instructions to the computer, the running process word RPW (first means of claim), priority level indicator word PRI (second means), J Table word and G Table word (third means), and micro-instructions of Figures 15c and 15c as found in the central store 1301. The invention resides in a "data structure" or an algorithm designed to control the multiprogramming computer to solve the particular problem indicated.

Under the ruling in *Gottschalk* v. *Benson*, 175 USPQ 673, the instant claims, depending upon a program implemented algorithm for patentability, are deemed nonstatutory subject matter.

3) All claims have been rejected. This action is hereby made a *FINAL REJECTION.*

CHARLES E. ATKINSON
EXAMINER
GROUP ART UNIT 236

LETTER TO OFFICE, DECEMBER 30, 1976.
AMENDMENT UNDER RULE 116

December 28, 1976

Commissioner of Patents & Trademarks
Washington
District of Columbia 20231

Dear Sir:

Responsive to the Office Action of October 27, 1976 and pursuant to Rule 116, please amend the above-identified application as follows:

REMARKS

Claims 1-6 are still active in the application.

Reconsideration of the instant application is respectfully requested.

Claims 1-6 have been rejected as being directed to non-statutory subject matter under 35 USC 100 and 101. The Examiner states in making this rejection as follows: "Although Applicants have now amended the claims to couch the claimed subject matter in language inferring structural entities and interconnections, the disclosed invention and specification language clearly indicates that the invention resides in a 'technique' or 'switch base instruction' for changing information in the system base both the main memory and internal scratchpad registers making the system base and scratchpad memory accessible to software but not visible to software i.e. cannot be read by the software. . . . The subject matter is deemed as the invention is then stated as 'a data structure'. . . . The term 'data structure' as recited in the claim is comprised of four entities described as 'first means', 'second means', 'third means', and 'fourth means'. Whether the so-called 'means' are hardware structure which store coded signals for performing a particular function or whether they refer to coded signals representing computer control words per se, any life which the claim may have in view of that which is immediately old in the art

resides in the 'technique' (page 5, line 4) or 'switch system base instruction' (page 5, lines 13 and 14) for solving the problem of changing information of the system base which is located in scratchpad registers which software cannot access (page 4, line 20)."

The Examiner then concludes in his discussion that "The invention then is embodied in the coded signals representing instructions to the computer, the running process word RPW (first means of claims), priority level indicator word PRI (second means), J table word and G table word (third means), and micro-instructions of Figures 15C and 15C(?) as found in the central store 1301. The invention resides in a 'data structure' or an algorithm designed to control the multiprogramming computer to solve the particular problem indicated."

Finally, the Examiner concludes "Under the ruling in Gottschalk v. Benson 175 USPQ 673, the instant claims, depending upon a program implemented algorithm for patentability, are deemed non-statutory subject matter."

It appears to the Applicants that the Examiner is using procrustean reasoning and forcing an arbitrary conclusion or solution to the facts. There appears to be a non-sequitur between the facts expounded and the conclusion reached. The claims are directed to data structures in hardware i.e. registers or computer memories. It can hardly be denied that registers and memories are hardware. The claims are directed to a specific new, novel and unobvious arrangement of these structures. This arrangement is more specifically recited in the claims and amplified in a discussion of Amendment A on pages 17-20 in this application.

More specifically, Claim 1 claims a data structure in main memory. Webster's Seventh New Collegiate Dictionary, published by G & C Merriam Company, Springfield, Massachusetts define a structure as:

(a) the elements of an entity or the position of such elements in their relationships to each other;

(b) the arrangements of particles or parts in a substance or body;

(c) something made up of interdependent parts in a definite pattern of organization;

(d) interrelation of parts is dominated by the general character of the whole.

The data structure system claimed is comprised of the first means which is the structure and arrangement of the running process word register at VAR+56 of the system base structure of Figure 6. The system base structure is described with the particularity of 35 USC 112 beginning on page 35 through 41. Its use is further described beginning on page 41. The running process word register structure is further shown with particularity on Figure 11J and is described with particularity beginning on page 39. The second means which is an element within the first means is part of the running process word register and is identified with reference numeral 1129. The third means within the data structure is the J table word and the G table word at BAR and BAR+4 on Figure 6 and further shown on Figures 11A and 11B and described in the specification. Finally, the fourth means claimed is shown in Figures 13A and 13B is a control store (i.e. ROM, etc.) as microprogrammed by the microprogram as disclosed by the flow charts of Figures 15B and 15C. Details of the control store which is microprogrammed is shown on Figure 13B (it should be understood that control stores are old in the art and persons of ordinary skill in the art can microprogram them when provided with a flow chart of the microprogram and this results in the firmware. It should be further noted that firmware should not be equated with microprograms per se. The Computer Dictionary and Handbook by Charles P. Sippl, et al, defines firmware as: "Logic circuits in read-only memory . . .)." Figures 13A and 13B are described with great particularity beginning on page 43 through page 53. Moreover, beginning on page 55 an example is taken by applying the microprogram flow chart of Figure 14A and actually microprogramming the control store of Figure 13B. Referring to Figure 14A and to page 55 of the specification, the first block 1402 of Figure 14A in-

structs the microprogrammer to fetch the IPQW word signals from the system base into scratchpad memory. Beginning on line 16 of page 55 of the specification, there is shown the generation of micro-operation signals 1360 and 1361 by hardware disclosed in accordance to 35 USC 112 to cause the retrieval of the IPQW word signals from the system base and storage subsystem 102 and transfer it to scratchpad memory 1350. Continuing further in the description of 14A there is similar descriptions with similar particularity to teach a person of ordinary skill in the art how to microprogram the control store or ROM in order to generate firmware, first of the dispatcher unit and more particularly the firmware of the instant invention. It should be noted that each and every means are hardware. The claimed language does not "*infer* structural entities and interconnections" as alleged by the Examiner, but actually claims structural entities and interconnections. The hardware handles and processes electrical signals, which is what computers do. The Examiner is incorrect when he states that the "disclosed invention and specification language clearly indicate that the invention resides in a 'technique' . . . ." The invention in any patent or patent application is what is claimed by the claim language. The specification may be used to interpret the bounds of the claim language. It appears that the Examiner is *clearly* disregarding the claim language and is urging upon us to interpret the invention strictly on the basis of "the specification language." This is contrary to accepted patent law. By reading the claim language, it will be seen that no "technique" is being claimed. Webster's Seventh New Collegiate Dictionary, by G & C Merriam Co., of Springfield, Massachusetts, published in 1969, define a technique as: "technical methods especially in scientific research"; "a method of accomplishing a desired aim." The Examiner is respectfully requested to reread the claim language and kindly inform the Applicants how hardware structure can arbitrarily be classified as a method or technique.

The Examiner also further alleges that "The disclosed invention and specification language clearly indicate that

the invention resides in 'a technique' or '*switch system base instruction*' for changing information in the system base in both the main memory and internal scratchpad registers making the system base in scratchpad memory accessible to software *but not visible to software i.e. cannot be read by the software.*" The apparent implication of this language is that instructions are techniques or methods and accordingly are unpatentable under 35 USC 100 and 101. It appears to the Applicants that the Examiner is confusing an algorithm of an instruction with the instruction itself. Instructions in computer programs have been patentable since time immemorial. It is well known in the computer art that what provides the power to a computer for accomplishing tasks are instructions. The more instructions a computer has, the greater power and functionality that it has to perform a variety of tasks. A machine capable of executing only two instructions is not as powerful as a machine capable of executing a hundred instructions; and it is obvious that these two machines are not the same machine. Instructions have been traditionally embodied in computers in three different ways as follows:

(a) by software programming i.e. (macroprogramming) of the computer (i.e. by placing the instruction capability temporarily in the computer).

(b) by microprogramming a control store (read only memory ROM) to generate firmware (i.e. by constructing the instruction capability to reside permanently in the computer; and,

(c) by building the instruction permanently in the computer in the form of hardware apparatus.

The first technique is the cheapest but also the slowest in operation; however, since it can be further programmed, alternatives can be made simply. The next technique, microprogramming, is a permanent embodiment in the control store (ROM memory) and is faster than the first technique but slightly more expensive. Finally, the last technique, building it into hardware, is the most expensive but it is very fast in execution. Today, almost

all major manufacturers utilize the second technique of embodying new functionality into a computer. The Applicants have disclosed the second technique (b)—i.e. firmware, as described supra. Firmware as previously noted supra has been defined by Sippl and Sippl as "logic circuits in read-only memory. . . ." Logic circuits are certainly patentable subject matter provided they meet the other requirements of patentability and there would be no question that logic circuits would not fall under 35 USC 100 or 101. Read only memories are also patentable subject matter and do not fall within 35 USC 100 or 101. Why then are novel logic circuits embodied in read only memories classified as nonpatentable subject matter? How can a piece of hardware be classified as a "technique" or a method?

It is submitted that the Examiner has erroneously interpreted the claims and classified the invention erroneously into a technique or a method. Even under this erroneous assumption, it would still not fall into the category of unpatentable subject matter because methods and techniques are patentable subject matter under 35 USC 100 and 101. It can only be inferred by the Applicants that what the Examiner was trying to establish is that the claimed subject matter is a mathematical algorithm and accordingly unpatentable under Gottschalk v. Benson & Tabbot. This, it is submitted, the Examiner has failed to do because in fact the invention is not a mathematical algorithm nor is it even an algorithm per se or even a method. Again the Applicants wish to stress it is hardware—that is what is disclosed and that is what is being claimed.

The Applicants would like to digress for a moment and submit that 35 USC 100 and 35 USC 101 were both included in the patent statutes to prevent individuals from trying to claim a scientific law, scientific principles, mathematical equations, pure mathematical algorithms, etc., which would preempt an entire scientific principle, mathematical formula, scientific law or method of doing business. It is obvious that merely because one discovers the law of gravity he should not be able to patent the law of gravity or that merely because one discovers relationships which pertain to basic fundamental laws

such as Einstein's $E=mc^2$ he should not be allowed to obtain a monopoly on the fundamental relationship or law itself via a patent. It is submitted, however, that inventions *based* on these scientific principles, mathematical equations, fundamental laws i.e. embodied into workable embodiments and utilizing such discoveries were never meant to be excluded by 35 USC 100 and 35 USC 101. (How many inventions have issued based on the law of gravity?) Yet this is what the Examiner in this instance is attempting to do—exclude a patentable invention merely because it is based on an algorithm. It is submitted that the Examiner is arbitrarily attempting to fit a hardware embodiment into a category in which it doesn't belong (i.e. a mathematical algorithm) so he can apply Benson & Tabbot and reject it on those grounds.

It is further submitted that at worst the Examiner has erroneously classified this invention as a *method* of operating a computer and at best that the invention *derives* from some general algorithm. Methods are patentable inventions provided they meet the criteria of patentability and should not categorically be classified as unpatentable subject matter. Moreover, inventions *derived* or *based* on scientific principles have been patentable since time immemorial. The majority or bulk of inventions that have been made since time immemorial derive from some scientific principles, basic scientific law, or mathematical algorithm. Surely the Examiner is not taking the position that all such inventions so derived are unpatentable. Moreover, merely because the scientific principle or theory on which the invention is based is described in the specification, does not negative the patentability of the embodiment of that scientific principle or theory. See In Re Storrs 114 USPQ 293 wherein it is stated "Applicant need not understand theory of operation of invention." See also In Re McCollough 136 USPQ 657 wherein it is stated "It is not customary in patent cases to require an Applicant either to know and disclose the underlying theory of his invention or to hold him to an expressed erroneous theory. Eames v. Andrews 122 US 40; Robinson on Patents, Paragraphs 485; In Re Modine 19 CCPA 1058, 57 Federal Second, 355, 13 USPQ 88. In principle,

the same would generally be true of an erroneous application o. theory, I think."

Accordingly, the Applicants should not be penalized merely because they disclose the scientific principle, algorithm, basic law or theory upon which their invention is based. Merely because the Applicants have gone the extra mile and disclosed the theory upon which their invention is based should not negative invention.

It is submitted therefore that the Applicants' invention is not, nor do the claims claim a theory, a mathematical algorithm, a scientific principle, or a fundamental physical law, and as a matter of fact, is not and does not claim a method of operating a computer. How then can the ruling in Gottschalk v. Benson 175 USPQ 673 apply? Several cases have spoken since Benson and Tabbot interpreting that case including In Re Christensen 178 USPQ 135, In Re Johnston 189 USPQ 257 and more recently the cases decided by the CCPA of In Re Noll and In Re Chatfield, all of which restrict the Benson and Tabbot decision to its own fact situation; that is the invention as claimed was unpatentable because it was claimed too broadly and, therefore, "would wholly preempt the mathematical formula" and practically "would be a patent on the algorithm itself." Judge Baldwin in In Re Noll said that claims define invention. Present claims (for In Re Noll) describe an apparatus, not a program. Claim must not be judged on a small abstracted portion deemed to be novel, but on the subject matter as a whole. The instant claims (in In Re Noll) are drawn to physical structure and not to an abstract "law of nature, a mathematical formula, or an algorithm." Appellants' machine is structurally different from a machine without a program. Moreover, instant claims (in In Re Noll) are limited to a particular technology, computer graphic systems and scan-conversion for graphic information. Furthermore, not all machines for achieving Appellants' results are included within the scope of Appellants' claims; non-program-controlled machines are clearly excluded. Court need not decide whether computer programs per se, constitute statutory subject matter. Moreover, Judge Baldwin further said that Benson does

not stand for a general proscription of patenting of computer programs and said that moreover, in review of Johnston, the Supreme Court emphasized that Benson was a limited holding.

In a nutshell (pardon the expression) then, Benson and Tabbot has been severely limited by subsequent interpretations of the CCPA. Benson and Tabbot stands for the proposition of claiming so broadly that the mathematical algorithm itself is claimed and thus there is preempted by the claim all uses of the mathematical algorithm. In the instant application it has been shown by a logical step by step progression that no mathematical algorithm is disclosed in the application nor claimed. As a matter of fact, no algorithm whatsoever is claimed. Moreover, no method or technique is claimed. What is claimed is strictly hardware; that whether or not this hardware embodiment claimed is developed from or based on a scientific principle, mathematical algorithm, law or nature, is immaterial as long as that mathematical algorithm, scientific principle or law of nature is not in itself claimed. It is submitted therefore that the reasoning of the Examiner in applying Benson and Tabbot is erroneous. It is erroneous both in fact and also in law. It is submitted that merely by calling something a mathematical algorithm, or a mathematical law, or a program does not make it so. Merely by placing a sign on a horse and saying "this is a cow" does not make that horse a cow, because the horse does not behave like a cow, and does not have the characteristics of a cow i.e. it does not have all the attributes of a cow, although it may have four legs, a tail, etc. To call hardware or firmware a program, or a mathematical algorithm, etc. does not make that firmware or hardware any more a program or mathematical algorithm because it does not have all the characteristics of a program or a mathematical algorithm. Accordingly, it is submitted that a procrustean solution does not stand up under the scrutiny of logic when attached either from the factual side or from the legal side. Accordingly, it is submitted that the instant invention as disclosed and claimed pertains to statutory subject matter under 35 USC 100 and 101 and

accordingly is allowable subject matter and allowance of Claims 1-6 is hereby respectfully requested and the application passed to issue at an early date.

If there is any way in which the application can be advanced to issue by a telephone call, the Examiner is requested to call the Applicants' Attorney collect.

Respectfully submitted,

/s/ Nicholas Prasinos
NICHOLAS PRASINOS
Attorney for Applicants
Registration No. 24,791

LETTER OF EXAMINER, FEBRUARY 1, 1977.

ADVISORY ACTION

[x] THE PERIOD FOR RESPONSE IS EXTENDED TO RUN 4 MONTHS FROM THE DATE OF THE FINAL REJECTION. 855 O. G. 1109.

[ ] Appellant's Brief is due in accordance with Rule 192 (a).

Applicant's response to the final rejection, filed Dec. 20, 1976, has been considered with the following effect, but it is not deemed to place the application in condition for allowance:

1. [ ] The proposed amendments to the claims and/or specification will not be entered and the final rejection stands because:

   a. [ ] There is no convincing showing under Rule 116 (b).
   b. [ ] They raise new issues that would require further consideration and/or search.
   c. [ ] They raise the issue of new matter.
   d. [ ] They are not deemed to place the application in better form for appeal by materially reducing or simplifying the issues for appeal.
   e. [ ] They present additional claims without cancelling a corresponding number of finally rejected claims.

2. [ ] Newly proposed or amended claims _____ would be allowed if submitted in a separately filed amendment cancelling the non-allowable claims.

3. [x] Upon the filing of an appeal, the proposed amendment [x] will be [ ] will not be, entered and the status of the claims in this application would be as follows:

   a. [ ] Claims _____ would be allowable.

   b. [x] Claims 1 - 6 would not be allowable.

   However: (See attachment for further explanation.)

   (1) [ ] The rejection of claims _____ on references is deemed to be overcome by applicant's response.

   (2) [ ] The rejection of claims _____ on non-reference grounds only is deemed to be overcome by applicant's response.

4. [ ] The affidavit, exhibit or request for reconsideration has been entered but does not overcome the rejection.

5. [ ] The affidavit or exhibit will not be admitted because applicant has not shown good and sufficient reasons why it was not earlier presented.

6. [ ] The application having been examined under the special accelerated examining procedure (M. P. E. P. 708.02), the proposed amendment has not been considered since it does not prima facie place the application in condition for allowance or in better condition for appeal.

Claims 1-6 are deemed properly rejected under 35 USC 101 as drawn to non-statutory subject matter. The U.S. Supreme Court in rendering a decision on the *Dann* v. *Johnston* case, 189 USPQ 257, held the invention to be obvious under 35 USC 103, thus reversing the CCPA. The Court did not render a decision on the 35 USC 101 issue and moreover did not modify its position taken in the *Benson* v. *Gottschalk* case.

In both the *In re Noll* and *In re Chatfield* cases the CCPA reversed the Board of Appeals decision. Until these decisions become final and since further review of clarification may be forthcoming, the PTO is continuing to examine computer program related applications in accordance with its current interpretation of the Supreme Court decision in the *Benson* case.

/s/ Charles E. Atkinson
CHARLES T. ATKINSON
EXAMINER
GROUP ART UNIT 236

BRIEF, APRIL 19, 1977.

JURISDICTION

This appeal is from a final rejection by the Primary Examiner of Group 236. The final rejection was dated October 27, 1976, and the Appellants then amended under Rule 116 on December 28, 1976, whereupon the Examiner gave an Advisory Action dated February 1, 1977 and extended the response period for 4 months from the date of the final rejection i.e. February 27, 1977. The Appellants filed a timely Notice of Appeal on February 21, 1977. Jurisdiction of this Board is invoked under the provisions of 35 USC Section 134 and CFR 1.191 of the Rules of Practice of the United States Patent Office in Patent Cases.

STATUS OF THE CLAIMS

Claims 1-6 are pending. Each of these six claims has been finally rejected and such rejection has been appealed.

CONCISE EXPLANATION OF THE INVENTION

This invention generally concerns a data processing system provided with a plurality of peripherals and having multiprocessing and multiprogramming capabilities. More specifically, the invention concerns a combination of hardware including a model-independent instruction means which is implemented in firmware, for adding capability and increased functionality to the computer system. As defined by the Applicants' claims, the present invention is directed to a combination of hardware structures controlled by a microprogrammed control system which provides for the generation of signals which cause the creation of an image of a communication center at another location, and for the generation of signals which causes the interchange of the information between the communication center and its image.

The computer system shown generally in Figure 1 is comprised of a main memory storage subsystem 102, a

processor subsystem 101, a peripheral subsystem 103. As shown on Figure 12, a portion of the main memory hardware is reserved for hardware use and system software use. A system base 1215 is defined as a communication center between the operating system and the various processes that may execute on the computer system. This is done in general by providing a BAR (Boundary Address Register) 1202 to address various entities in the system base 1215 which then can address by indirection any number of process control blocks, PCB's 1206, each processing control block being associated with a particular process that can execute in the system. Accordingly, the computer hardware (i.e. registers, scratchpad memory 1315, peripherals, etc.) and the operating system (not shown, but described in reference 12, page 3 of the specification and incorporated by reference therein) can communicate directly with any of myriads of processes that can execute on the computer, without programmer intervention. The central processing unit CPU is comprised generally as shown on Figure 13a of an arithmetic logic unit ALU 1317, the instruction fetch unit IFU 1318, the address control unit ACU 1319, several general, base and scientific registers and scratchpad memory. The control unit including formats of microwords and the logic hardware for generating control signals in response to the microwords and microprogramming, is shown in Figure 13b. Associated with the ALU 1317 and the CSU 1301, is a scratchpad memory unit LSU 1315 (sometimes referred to also as the local store unit). The operations of the arithmetic logic unit are completely dependent on micro-operation control signals from the control store unit shown as CSU 1301 and Figure 13b. Scratchpad memory 1315 is used to store CPU control information and maintainability information. In addition, the scratchpad memory 1315 contains working locations which are primarily used for temporary storage of operands and partial results during the data manipulation.

With respect to this invention, the system base, which is located at a location specified by the boundary address register BAR in main memory, is copied into a new loca-

tion in main memory with predetermined portions of its information changed. This is readily accomplished by the operating system. (See application reference 12, on page 3, of the specification incorporated by reference into the application herein). The switch system base instruction then transfers the information which is in the system base at the new location into the system base at the location specified by BAR and also that portion of the system located in scratchpad memory. As defined by Applicants' claims, this invention is directed to hardware structures controlled by a microprogrammed control system Figure 13b, which provides for the generation of signals which cause the creation of an image of a communication center at another location and for the generation of signals which causes the interchange of the information between the communication center and its image. (See Claim 2). This communication system is provided by the claimed combination which causes an image of the communication center to be generated at another location in main memory, wherein the information in the image is then updated and then retransferred back to the communication center and to the scratchpad memory.

Claim 1 claims the means for storing coded signals of addresses and priority to address any selected one of the processes, for generating signals which cause the exchange of information between the claimed data structures and said scratchpad registers.

Claim 2 claims means for generating signals which cause the creation of an image in the communication center and claims means for generating signals causing an exchange of information within the created image and further claims means for generating signals which cause a transfer of the new image to the communication center and to the scratchpad memory.

Claim 3 further limits and defines the third means of Claim 2.

Claim 4 includes in the claimed combination the arithmetic and logic unit having an auxiliary memory.

Claim 5 claims additional hardware—the hardware gate.

Claim 6 further limits Claim 5.

* * * *

For Claims 1 to 6 inclusive, see Rejected Claims 1 to 6 inclusive.

REJECTION

Claims 1-6 have been rejected finally under 35 USC 101 as drawn to non-statutory subject matter. The Examiner states "The U.S. Supreme Court in rendering a decision on the Dann v. Johnston, 189 USPQ 257, held the invention to be obvious under 35 USC 103, thus reversing the CCPA. The Court did not render a decision on the 35 USC 101 issue and moreover did not modify its position taken in the Benson v. Gottschalk case.

In both the In Re Knoll and In Re Chatfield cases the CCPA reversed the Board of Appeals decision. Until these decisions become final and since further review of clarification may be forthcoming, the PTO is continuing to examine computer programs related application in accordance with its current interpretation of the Supreme Court decision in the Benson case."

ARGUMENT

It appears to the Appellant that the Examiner is using procrustean reasoning and forcing an arbitrary conclusion or solution to the facts. There appears to be a non-sequitur between the facts expounded and the conclusion reached. The claims are directed to a combination of hardware and firmware i.e. registers or computer memories etc. It can hardly be denied that registers and memories are hardware. The claims are directed to a specific new, novel and unobvious arrangement of these structures. This arrangement is more specifically recited in the claims and amplified in a discussion on pages 17-20 in this application.

Claim 1 specifically claims a data structure in main memory. Webster's Seventh New Collegiate Dictionary, published by G & C Merriam Company of Springfield, Massachusetts, define a structure as:

(a) The elements of an entity of a position of such elements in their relationship to each other;

(b) The arrangements of particles or parts in a substance or body;

(c) Something made up of interdependent parts in a definite pattern of organization;

(d) Interrelationships of parts as dominated by the general character of the whole.

A data structure claimed is comprised of the first means which is the structure of the running process word register at BAR + 56 of the system base structure of Figure 6. The system base structure is described with the particularity of 35 USC 112 beginning on page 35 through 41. Its use is further described beginning on page 41. The running process word register structure is further shown with particularity on Figure 11j and is described with particularity beginning on page 39 of the specification. The second means which is an element within the first is part of the running process word register and is identified with reference to numeral 1129. The third means within the data structure is the J table word means and the G table word means at BAR and BAR + 4 on Figures 11a and 11b and described in the specification. Finally, the fourth means claimed is shown on Figure 13a and 13b is a central store as microprogrammed by the microprogram as disclosed by the flow charts of Figures 15b and 15c. Details of the control store which is microprogrammed are shown on Figure 13b (it should be understood that control stores are old in the art and persons of ordinary skill in the art can microprogram them when provided with the flow charts of the microprogram and this results in the firmware. It should be further noted that firmware should not be equated with the microprogram per se. The Computer Dictionary and Handbook, by Sippl and Sippl, defines firmware as "logic circuits in read-only memory . . .)." Figures 13a and 13b are described with great particularity beginning on page 43 through 53 of the specifica-

tion. Moreover, beginning on page 55, an example is taken by applying the microprogram flow chart of Figure 14a and actually microprogramming the control store of Figure 13b to generate the signals for controlling certain predetermined tasks and functions. For example, referring to Figures 13b and 14a and to page 55 of the specification, the first block 1402 of Figure 14a instructs the hardware of Figure 13b to generate signals to fetch the IPQW word from the system base into scratchpad memory. (See sample generation of such signals by the hardware of Figure 13b). Beginning on line 16 of page 55 of the specification, there is shown the generation of the micro-operation signal 1360, 1361 to cause the retrieval of the IPQW word from the system base and storage subsystem 102, and transfer it to scratchpad memory 1315. Continuing further in the description of 14a, there is similar descriptions with similar particularity to teach a person of ordinary skill in the art how to microprogram the control or ROM in order to generate the firmware, first of the dispatch unit and more particularly the firmware of the instant invention. It should be noted that every means claimed is hardware. The claimed language does not "infer structural entities and interconnections" as alleged by the Examiner, but actually claims structural entities and interconnections. The hardware handles and processes electrical signals, which is what computers do. The Examiner is incorrect when he states that the "disclosed invention and specification language clearly indicates that the invention resides in a 'technique'." The invention in any patent or patent application is what is claimed by the claimed language. The specification may be used to interrupt the bounds of the claimed language. It appears that the Examiner is clearly disregarding the claimed language and is urging upon us to interpret the invention strictly on the basis of his interpretation of the specification language." This is contrary to accepted patent law. By reading the claimed language, it will be seen that no "technique" is being claimed. Webster's Seventh New Collegiate Dictionary, by G & C Merriam Co. of Springfield, Massachusetts, published in 1969, de-

fine a technique as "technical methods especially in scientific research"; "a method of accomplishing a desired aim." The Examiner is clearly wrong in arbitrarily classifying hardware structure as a method or technique.

The Examiner also further alleges that "the disclosed invention and specification language clearly indicates that the invention resides in 'a technique' or 'switch system base instruction' for changing information in the system base in both the main memory and internal scratchpad registers. Changing the system base in scratchpad memory accessible to software but not visible to software i.e. cannot be read by the software." The apparent implications of this language is that instructions are techniques or methods and accordingly are unpatentable under 35 USC 100 and 101. It appears to the Applicants that the Examiner is confusing an algorithm of an instruction with an instruction itself embodied in some medium. Instructions embodied in computer systems have been patentable since time immemorial. It is well known in the computer art that what provides the power to a computer for accomplishing tasks are instructions. The more instructions a computer has, the greater power and functionality that it has to perform a variety of tasks. A machine capable of executing only two instructions is not as powerful as a machine capable of executing 100 instructions; and it is obvious that these two machines are not the same machine. Instructions have been traditionally embodied in computers in three different ways as follows:

(a) By software programming i.e. (macroprogramming of the computer), i.e. by placing the instruction capability temporarily in the computer (i.e. in evanescent main memory form).

(b) By microprogramming a control store (read only memory ROM) to generate firmware (i.e. by constructing the instruction capability to reside permanently in the computer (i.e. logic circuits in ROM memory, generally in the form of integrated circuits); and,

(c) By building the instruction permanently in the computer in the form of hardware apparatus (i.e. discrete logic circuits).

The first technique is the cheapest but also the slowest in operation; however, since this can be further programmed and easily changed, alterations can be made simply. The next technique, microprogramming, is a permanent embodiment in the control store (ROM) memory and is faster than the first technique but slightly more expensive.

Finally, the last technique, building it into hardware, is the most expensive but it is very fast in execution. Today, almost all major manufacturers utilize the second technique (b) of embodying new functionality into computers. The Applicants have disclosed the second technique (b)—i.e. firmware, as described supra. Firmware as previously noted supra has been defined by Sippl and Sippl in their authoritative Computer Dictionary and Handbook as "logic circuits in read only memory . . . ." Large scale integration (LSI) of circuits have made this possible. Logic circuits are certainly patentable subject matter provided they meet the other requirements of patentability and there would be no question that logic circuits would not fall under 35 USC 100 or 101. Read only memories are also patentable subject matter and do not fall within 35 USC 100 or 101. Why then are novel logic circuits embodied in read only memories classified as non-patentable subject matter? How can a piece of hardware be classified as a "technique" or a "method"?

It is submitted that the Examiner has erroneously interpreted the claims and classified the invention erroneously into a technique or a method. Even under this erroneous assumption, it would still not fall into the category of unpatentable subject matter under 35 USC 100 and 101 because methods and techniques are patentable subject matter under 35 USC 100 and 101. See the following cases:

(a) In Re Abrams (89 USPQ 266) 1951;
(b) In Re Shao Wen Yua (89 USPQ 324) 1951;

(c) In Re Jones (153 USPQ 77) 1967;
(d) In Re Tarczy-Hornoch (158 USPQ 141) 1968;
(e) In Re Praeter and Wei (102 USPQ 541) 1969;
(f) In Re Bernhart and Fetter (163 USPQ 611) 1969;
(g) In Re Mahoney (164 USPQ 572) 1970;
(h) In Re Musgrave (167 USPQ 280) 1970;
(i) In Re Foster, et al (169 USPQ 99) 1971;
(j) In Re McIlroy (170 USPQ 430) 1972;
(k) In Re Waldbaum (173 USPQ 430) 1972;
(l) In Re Christensen (178 USPQ 35) 1973.

Omitted from the above cases, all of which held that programs were patentable subject matter, is the Benson and Tabbot decision since subsequent decisions of the Supreme Court (i.e. In Re Johnston 183 USPQ 172) and the CCPA (i.e. In Re Christensen 178 USPQ 35, and in In Re Noll 191 USPQ 271), and In Re Chatfield 191 USPQ 73), have seriously limited the Benson & Tabbot decision to its own fact situation of overclaiming so that it "would wholly preempt the mathematical formula", and particularly "would be a patent on the algorithm itself."

It can only be inferred by the Applicants that what the Examiner was trying to establish is that the claimed subject matter is a mathematical algorithm and accordingly unpatentable under Gottschalk v. Benson and Tabbot. This, it is submitted, the Examiner has failed to do because in fact, the invention is not a mathematical algorithm, nor is it even an algorithm per se, or even a method. Again, the Applicants wish to stress it is a combination of hardware and firmware—that is what is disclosed in the specification and that is what is being claimed.

The Applicants would like to digress for a moment and submit that 35 USC 100 and 35 USC 101 were both included in the patent statutes to prevent individuals from trying to claim a scientific law, scientific principles, mathematical equations, pure mathematical algorithms, methods of doing business, etc., which would preempt an entire scientific principle, mathematical formula, scien-

tific law or method of doing business. It is obvious that merely because one discovers the law of gravity, he should not be able to patent the law of gravity or that merely because one discovers relationships which pertain to basic fundamental laws such as Einstein's $E = mc^2$, he should not be allowed to obtain a monopoly on the fundamental relationships alone or law itself via a patent. It is submitted, however, that inventions *based* on these scientific principles, mathematical equations, fundamental laws, i.e. embodied into workable embodiments and utilizing such discoveries were never meant to be excluded by 35 USC 100 and 35 USC 101. (How many inventions have issues based on the law of gravity)? Yet this is what the Examiner in this instance is attempting to do—exclude a patentable invention merely because it is based on an algorithm. It is submitted that the Examiner is arbitrarily attempting to fit a hardware embodiment into a category in which it doesn't belong (i.e. a mathematical algorithm) so he can apply Benson & Tabbot and reject it on those grounds. If this were permitted, then hardly any computer related invention would be patentable since most computer system inventions are based on some form of an algorithm. As noted before, this algorithm may be implemented in software, firmware or hardware. It is submitted that Benson & Tabbot was finally rejected because the invention claimed would preempt a mathematical algorithm and would be a monopoly on the algorithm itself. Nor was the claimed invention in Benson & Tabbot restricted with respect to application or field of endeavor.

It is further submitted that at worst the Examiner has erroneously classified this invention as a *method* of operating a computer and at best that the invention *derived* from some general algorithm. Methods are patentable inventions provided they meet the criteria of patentability and they should not be categorically classified as unpatentable subject matter. Moreover, inventions *derived* or *based* on a scientific principle, have been patentable since time immemorial. The majority or bulk of inventions that have been made since time immemorial derive from some scientific principle, basic scientific law, or

mathematical algorithm. Surely the Examiner is not taking the position that all such inventions so derived are unpatentable. Moreover, merely because the scientific principle or theory of which the invention is based is described in the specification, does not negative the patentability of the embodiment of that scientific principle or theory to a hardware invention. See In Re Storrs 114 USPQ 293 wherein it is stated: "Applicant need not understand theory of operation of invention." See also In Re McCullough 136 USPQ 657 wherein it is stated: "It is not customary in patent cases to require an Applicant either to know and disclose the underlying theory of his invention, or to hold him to an express erroneous theory. Eames v. Andrews 122 US 40; Robinson on Patents, Paragraph 485; In Re Modine 19 CCPA 1058; 57 Federal Second, 355, 13 USPQ 88. In principle, the same would generally be true of an erroneous application of theory I think." Accordingly, the Applicants should not be penalized merely because they disclose the scientific principle, algorithm, basic law or theory upon which their invention is based. Merely because the Applicants have gone the extra mile and disclosed the theory upon which their invention is based, should not negative invention.

It is submitted therefore that the Applicants' invention is not, nor do the claims claim a theory, a mathematical algorithm, a scientific principle, or a fundamental physical law, and as a matter of fact, is not and does not claim a method of operating a computer. How then can the decision in Gottschalk v. Benson 155 USPQ 673 apply in the instant invention? Several cases have spoken since Benson & Tabbot interpreting that case including In Re Christensen 158 USPQ 135, In Re Johnston 189 USPQ 257 and more recently the cases decided by the CCPA, In Re Noll and In Re Chatfield (citations supra), all of which restrict the Benson & Tabbot decision to its own fact situation; that is the invention as claimed in Benson & Tabbot was unpatentable because it was claimed too broadly, and therefore, "would wholly *preempt the mathematical formula*" and practically "would be a *patent on the algorithm itself*." Judge Baldwin in In Re Noll said that claims define invention. The present claims (for

In Re Noll) describe an apparatus, not a program. Claims must not be judged on a small abstracted portion deemed to be novel, but on the subject matter as a whole. The instant claims (In Re Noll) are drawn to physical structure and not to an abstract "law of nature, a mathematical formula, or an algorithm." Appellants' machine is structurally different from a machine without a program. Moreover, instant claims (in In Re Noll) are limited to a particular technology, computer graphic systems and scan-conversion for graphic information. Furthermore, not all machines for achieving Appellants' results are included within the scope of Appellants' claims; non-programmed controlled machines are clearly excluded. The Courts need not decide whether computer programs per se, constitute statutory subject. Moreover, Judge Baldwin further said that Benson does not stand for a general proscription of patenting of computer programs and said that moreover, in review of Johnston, the Supreme Court emphasized that Benson was a limited holding.

In a nutshell, (pardon the expression) then, Benson & Tabbot has been severely limited by subsequent interpretations of the CCPA and by its own decision in In Re Johnston, Benson & Tabbot stand for the proposition of claiming so broadly that the mathematical algorithm itself is claimed and thus there is preempted by the claims all uses of the *mathematical algorithm.* In the instant application it has been shown by a logical step by step progression that no mathematical algorithm is disclosed in the application nor claimed. As a matter of fact, no algorithm whatsoever is claimed. Moreover, no method or technique is claimed. What is claimed is a combination comprised of strictly hardware; that whether or not this hardware embodiment claimed is developed from or based on a scientific principle, mathematical algorithm, law of nature is immaterial as long as that mathematical algorithm, scientific principle or law of nature is not *in itself* claimed.

Nor is a program claimed by the instant invention. Moreover, it is submitted that the claimed language is not a subterfuge for patenting a program. Note that In

Re Noll and In Re Chatfield, the subject matter was acknowledged to be a program—not firmware, and not hardware. It is submitted that the instant invention has been attacked merely because of guilt by association—by the company it keeps—in that it is computer related, and takes advantage of the latest advances in the computer art for its embodiment. This statement is given added weight by the Examiner's statement in his Advisory Action of 2/1/77 wherein he states:

"In both the In Re Noll and In Re Chatfield cases, the CCPA reversed the Board of Appeals' decision. Until these decisions become final and since further review of clarification may be forthcoming, the PTO is continuing to examine *computer program related applications* in accordance with its current interpretations of the Supreme Court decisions in the Benson case." (Underlining added).

The theory of guilt by association has long been discredited. It should not be revived now in one of the most progressive agencies of the U.S. Government by procrustean reasoning. It is submitted, therefore, that the reasoning of the Examiner in applying Benson & Tabbot is erroneous. It is erroneous both in fact and also in law. It is submitted that merely by calling something a mathematical algorithm or a mathematical law, or a program does not make it so. Merely by placing a sign on a horse and saying "this is a cow" does not make that horse a cow, because the horse does not behave like a cow, and does not have the characteristics of a cow, i.e. does not have all the attributes of a cow, although it may have four legs, a tail, etc., and in many ways may be similar to a cow. To call hardware or firmware a program, mathematical algorithm, etc., does not make that firmware or hardware any more a program or mathematical algorithm because it does not have all the characteristics of a program or a mathematical algorithm. Accordingly, it is submitted that the proscrustean solution of the Examiner does not stand up under the scrutiny of logic when attacked either from the factual side or from the legal side. Accordingly, it is submitted that the instant invention as disclosed and claimed pertains to statu-

tory subject matter under 35 USC 100 and 101 and accordingly is allowable subject matter.

CONCLUSION

Appellants respectfully submit that the claims in issue have been shown to be clearly allowable subject matter which is and always has been patentable under 35 USC 100 and 35 USC 101. Therefore, Appellants respectfully request that the Examiner's rejection of the appealed claims be reversed and that the application be allowed.

ORAL HEARING ON REQUEST

Appellants hereby request an Oral Hearing. Two extra copies of this Brief are provided in accordance with Rule 192.

Respectfully submitted,

/s/ Nicholas Prasinos
NICHOLAS PRASINOS
Attorney for Appellants
Registration No. 24,791

EXAMINER'S ANSWER, AUGUST 4, 1977.

This is an appeal from the final rejection of claims 1 through 6, all the claims remaining in the application.

A correct copy of the claims appear on pages 6 through 8 of applicants' brief.

STATEMENT OF GROUND OF REJECTION:

Claims 1 through 6 are rejected under 35 U.S.C. 101 as drawn to non-statutory subject matter under the ruling of the Supreme Court decision, Gottschalk v. Benson, 409 US 63, 175 USPQ 673.

In a subsequent Court decision, *In re Johnston*, 582 F.2d 765, 183 USPQ 172 (CCPA 1974), the Court held the claims unpatentable under 35 USC and did not render a decision on the 35 USC 101 rejection. Moreover, the Court did not modify its position taken in the *Benson* case.

On November 18, 1976, the Court of Claims and Patent Appeals decided the cases of *In re Noll* and *In re Chatfield*. In both cases the Board of Appeals had taken the position that the claims were drawn to non-statutory subject matter under the *Benson* decision. The CCPA reversed the Board of Appeals in both cases. However, per O.G. Notice 954 O.G. 550 dated January 11, 1977, until these decisions become final and since Request for Certiorari has been filed in both cases, the patent examining corps is directed to continue to examine computer programming applications in accordance with its current interpretation of the Supreme Court decision in the *Benson* case.

Applicant has set forth the background and general field of the invention beginning with page 3 of the application. The general field of invention is stated as that of a system and method for communicating between hardware and software in a multiprogramming/multiprocessing computer environment. It is then stated that multiprogramming/multiprocessing computer systems re-

quire the use of a communications area in main storage for communications between hardware and software. Applicants further state that high performance computing systems find it necessary to store portions of the system base in internal scratchpad registers instead of main memory and consequently the portion of the system base in the scratchpad registers is inaccessible and invisible to software running in the system. The problem to which this invention is directed was stated by applicants as the condition that arises when "software (i.e. the programmer) wants to change positions or change information of the system base which happens to be located in these internal central processor scratchpad registers which software cannot access". Applicants proceed to discuss two prior art methods or techniques to solve the problem and states their disadvantages and shortcomings. On page 5 of the specification, the objectives of the instant invention are set forth and include such statements as "a primary object of the invention is to provide an improved system and method for changing information in the system based of a computer system", "another object of the invention is to provide a switch-system base instruction in firmware which changes, upon execution, information in the system base". Thus, applicants have established the field of the invention, the problem to be solved, and have further given an indication of that which is deemed old in the art and that which is to be deemed their contribution to the art. Indeed, the claims define the invention but the claims must be interpreted in light of the disclosure.

In the amendment filed August 11, 1976, an Appendix I is set forth to provide further background of the invention and is specifically addressed to the first Office Action rejection wherein all the claims were rejected as drawn to inadequately disclosed subject matter. In this appendix, applicants set forth publications and U.S. patents as documented evidence that certain systems and devices required to carry out the invention in the instant application are well known in the data processing art. The examiner agrees that these publications and patents do point out the fact that the operating system within which

applicants' invention is utilized is well known in the art. There can be no controversy it would appear that the prior art does comprise multiprogramming/multiprocessing computer system comprising a CPU, main memory scratchpad registers, etc. and means for communicating between the operating system and the processes.

Applicants have stated in the brief, page 9, that "claim 1 specifically claims a data structure in main memory". It would appear then that applicants and the examiner agree as to the nature of applicants' contribution to the art and that the portion of claim 1 which sets forth the subject matter deemed to distinguish over that which is old in the art is the portion defining the "data structure". Applicants further state in the brief, page 13, that they have chosen to implement their improvement of an existing operating computer system by the second of three "techniques". This second technique of implementing "instructions" into a computer is generally categorized as "firmware" as opposed to software programming and hardwired apparatus. Applicants contend that their "firmware" invention when loaded into the registers of the operating system result in an invention having structural aspects and hardware limitations. Claim 1 has been couched in apparatus format and recites first means through fourth means. The first means through third means set forth means for storing signals and read upon registers or memory locations within the operating system, already shown to be old in the art. The function of or attribute of the signals stored in the first, second, and third means for storing may be classified as indicating an address for a selected one of a plurality of processes, indicating a priority of said selected process relative to other processes, and indicating an address of a selected one of the plurality of processes, respectively. There is clearly no novelty to be found here for distinguishing over that which is deemed to be old in the art.

The fourth means is stated as means for generating signals causing the changing of information in the data structure and scratchpad registers and is categorized by applicants' brief, page 10, as a control store as micropro-

grammed by the microprogram of Figures 15b and 15c. The use of control stores in computer systems has been shown to be old in the art in the listing of publications and U. S. Patents in paragraph 19 of Appendix I, amendment filed August 11, 1976. In fact, paragraph 22 of the Appendix I correctly indicates that microprogrammed computer processors including microprogrammed control units are old in the art and suggests that the systems shown in the two cited patents are capable of practicing or implementing the instant invention.

Applicants' contention that *Benson* is not properly applied against the claimed invention because of the hardware limitations expressed in the claims cannot stand. As correctly stated in the objectives of the invention within the disclosure and as pointed out in the previous discussion of the claim recitation in light of that which is deemed old in the art, the invention resides in a computer instruction in firmware. The so called limitations expressed in claim 1 are so conventional that it would have been the exceptional computer at the time of applicants' filing date which would not include such limitations. Applicants' invention resides in an instruction in firmware which changes, upon execution, information in the system base. The claim is thus reciting prior art coupled with subject matter which the U.S. Supreme Court has found to be non-statutory in *Benson*. As stated by Justice Douglas in *Benson*:

> "It may be that the patent laws should be extended to cover these programs, a policy matter to which we are not competent to speak.
>
> * * * *
>
> "If these programs are to be patentable, considerable problems are raised which only committees of Congress can manage, for broad powers of investigation are needed, including hearings which canvass the wide variety of views which those operating in this field entertain. The technological problems tendered red in the many briefs before us indicate to us that considered action by the Congress is needed".

Whereas the previous discussion has been limited to claim 1, the same rationale applies equally to claim 2. The recitation of claim 2 perhaps more clearly points out the contribution of applicants to the prior art. The three means recited as comprising the improvement are recited as means for generating signals to cause the computer to perform certain functions. These means must be construed to read upon memory locations comprising instructions required to command and control the computer elements for performing the functions recited. Applicants' contribution to the art resides in the specific instructions found within the memory locations and not the combination of three memory locations labeled first, second, and third means.

The examiner must take exception to the statement on page 14 of the brief wherein the list of cases labeled (a) through (b) are referred to as holding that programs are patentable subject matter. The examiner has read most, if not all, of the above cases at some point in time and must disagree vehemently with applicants' observation as to their holding that computer programs are statutory subject matter under 35 U.S.C. 101. The examiner's observation would be that none of the listed cases hold that computer programs are statutory subject matter.

The examiner respectfully submits that claims 1 through 6 are properly rejected as drawn to non-statutory subject matter under 35 U.S.C. 101.

/s/ Charles E. Atkinson
CHARLES E. ATKINSON
EXAMINER
GROUP ART UNIT 236

CONFEREE: M. A. MORRISON

06/28/77

REPLY BRIEF, AUGUST 17, 1977.

GENERAL

This Reply Brief is necessitated because the Examiner has raised new points and new issues in his answer of August 4, 1977. Under 37 CFR 1.193(b) the Appellant has a right to file a Reply Brief within twenty days from the date of Examiner's answer directed to such new points or arguments as have been raised by the Examiner's answer.

NEW POINTS AND ARGUMENTS RAISED BY THE EXAMINER

The Examiner's answer of August 4, 1977 attacks the elements of the Appellants' claims as being old in the art without citing references and without applying anticipation or obviousness rejections; accordingly no weight should be given to such arguments. Since the Examiner has not applied either a 35 USC 102 or a 35 USC 103 rejection to the claims, it is respectfully requested that all references to elements that are old in the art, and by inference anticipated or obvious, be stricken from the record, since neither anticipation or obviousness under 35 USC 102 or 35 USC 103 is an issue in the instant appeal. Examples of such references by the Examiner to elements being old in the art are as follows:

1. Page 5, beginning on line 6, it is stated "Claim 1 has been couched in apparatus format and recites first means through fourth means. The first means through third means set forth means for storing signals and read upon registers or memory locations within the operating system, already shown to be all old in the art . . . there is clearly no novelty to be found here by distinguishing over that which is deemed to be old in the art."

2. On page 5, beginning on the 6th line from the bottom, it is stated "The fourth means is stated as means for generating signals causing the changing of information in the data structure and scratchpad registers and is categorized by Applicants' Brief, page 10, as a control store as microprogrammed by the microprogram of Fig-

ures 15b and 15c. The use of control stores in computer systems has been shown to be old in the art. . . ."

3. Page 6, beginning on the tenth line from the bottom it is stated as follows "The so-called limitations expressed in Claim 1 are so conventional they would have been the exceptional computer at the time of Applicants' filing date which would not include such limitations."

Finally, the Examiner states on page 7, beginning at line 8 from the bottom as follows "Applicants' contribution to the art resides in the specific instructions found within the memory locations and not the combination of three memory locations labeled first, second and third means."

Thus, the Examiner appears to be utilizing, anticipation and obviousness arguments, not issues on this appeal and without citing references, in an attempt to delineate what is new and what is old in the claim; thus having dismembered of claim (contrary to the adjudicated decisions to be discussed infra), attempts to fit the claim into a procrustean bed of 35 USC 101, instead of 35 USC 102 or 103 wherein all the facts fit.

It is submitted that by such procrustean reasoning, objects may be dissected to a point where each element of each object can be categorized for example as an arm, a leg, an eye, a tail, etc. Yet when one then tries to resynthesize the original object from an aggregate of such categorized objects as arms, legs, eyes, tails, taking any one arm or leg, etc. of various objects in accordance to category but without distinguishing the source, one doesn't obtain the original object. In a similar vein, to focus attention on only one of a claim element, as the Examiner has done and to lightly dismiss the others as inconsequential to Applicants' contribution to the prior art, is to mischaracterize the claim. The Applicants' contribution to the prior art is correctly interpreted and delineated from the prior art only when the subject matter is taken as a whole. (See infra). Only when all the elements of a claim, and how they relate, cooperate and function with each other to solve a particular problem, are considered as a whole, can a claim be properly evalu-

ated; and it makes no difference whether such elements are individually old in the art.

ARGUMENTS

Since there are no issues pertaining to anticipation or obviousness, the Appellants need not address this issue at the present time but will address the issue of whether or not it is proper to dismember claims and focus only on a small portion of the claim without treating the subject matter of the claim as a whole.

In In Re Wynne and Cousen, 118 USPQ 306, the Court stated "It is immaterial that claimed steps may be old individually where, by the claimed combination of steps, Applicant solves problem and produces desirable results not heretofore obtainable . . . claim is allowed since it cannot be said that soundness of rejection is clear beyond doubt; doubts in such cases are resolved in favor of the Applicant."

In In Re Warner and Warner, 154 USPQ 173, it is stated "Patentability of an invention is not viewed with hindsight; whether individual elements of combination are old begs the question; thus, where invention sought to be patented resides in combination of old elements, inquiry is whether bringing them together was obvious, not whether one of ordinary skill, having the invention before him would find it obvious through hindsight to construct the invention from elements of prior art." In that same case it is further stated "Language of 35 USC 102 that a person shall be entitled to a patent unless, concerning novelty and unobviousness, places burden of proof on Patent Office to produce factual bases for its rejection under Section 102 and 103; where such proof is lacking there is no necessity for resolving doubt in favor of Office's position."

In In Re Aufhauser, 158 USPQ 351, the Court said "*Concept of 'invention as a whole' comprises the total inventive act; concept includes not only the remedy claimed, but also the discovery of the source of trouble despite teachings of prior art; when considering what constitutes Applicant's 'invention as a whole' within con-*

text of 35 USC 102, Court must consider first that which Applicant discovered as cause of trouble which persisted despite the prior art, and then consider Applicant's remedy." (underlining added)

In a more recent decision published in the 11/18/76 issue of BNA's Patent, Trademark & Copyright Journal, reporting on Clopsay Corp. v. Blessings Corp., decided 10/21/76, it was stated "While the creation of a plastic film embossed with a perpendicular design may not seem like a 'substantial advance' over a previous film embossed with a diagonal design, the discovery that the new design prevents the edges of the film from curling represents a patentable advance in the eyes of the U.S. District Court for Delaware. 'It is a well established principle' says Judge Stapleton, 'that one may add to the sum of useful knowledge by uncovering the nonobvious source of a problem and in devising a solution which, though involving a combination of *old elements obvious after the source of the problem is isolated,* has never before occurred to those skilled in the art.' "

In In Re Warner and Warner, 154 USPQ 173, it is stated "Patentability of invention is not viewed with hindsight; whether individual elements of combination are old begs the question; thus, where invention sought to be patented resides in combination of old elements, inquiry is whether bringing them together was obvious. . . ."

Clearly the adjudicated decisions establishes that the subject matter of a claim must be treated as a whole and in some instances include the recognition of a problem which had eluded the prior art. Clearly the Examiner has not done this. He has dissected the claim language and eliminated the majority of elements in order to force-fit the invention into a category he believes the invention should be in and not to the category that the invention actually is in, as defined by the claims in light of the specification. It is submitted that his interpretation of the claim is erroneous. Moreover, the Appellants do not agree with him, as stated on page 4, line 19, of the Examiner's answer that "It would appear then that Applicants and the Examiner agree as to the nature of Applicants'

contribution to the art and that the portion of Claim 1 which sets forth the subject matter deemed to distinguish over that which is old in the art is the portion defining the 'data structure.'" The elements of the first claim are discussed in the Brief for the Appellant, beginning on line 21, of page 9, through line 27 of page 11. Clearly, nothing but hardware is claimed in all of Claim 1. Whether the structure claimed is old begs the question since anticipation or novelty are not issues on this appeal. Similarly each and every claim remaining in the application claims nothing but hardware. The contribution of the Applicants' invention to the prior art is clearly and succinctly set forth in the Brief for the Appellant, beginning at the bottom line of page 3 through and including line 6 of page 5 and moreover such contribution is further delineated in the claims.

The Examiner further mischaracterizes the nature of the Applicants' invention on page 7, beginning on line 20 wherein he states "The three means recited as comprising the improvement are recited as means for generating signals to cause the computer to perform certain functions. These means must be construed to read upon memory locations comprising instructions required to command and control the computer elements for performing the function recited. Applicants' contribution to the art resides in the specific instructions found within the memory locations and not the combinations of three memory locations labeled first, second and third means." It is abundantly clear that memory locations do not generate signals. What generates signals is typically the hardware depicted in Figure 13b. The micro-operation signals of Figure 13b are designated by numerals 360-365 and are mainly generated by the control unit (CSU 1301), the control store array 1333, the micro-op decoder 1359, and the logic circuitry designated α through ω. This is all hardware. A permanently microprogrammed ROM (read only memory) in the control store array 1333 is known as the firmware. It is typically comprised of large scale integrated (LSI) semiconductory circuitry. Firmware is defined in the *Computer Dictionary and Handbook*, by Charles J. Sippl and Charles P. Sippl on page 186 as "logic circuits in read-only memory. . . ."

Figures 13a and 13b are described with great particularity beginning on page 43 through 53 of the specification. Moreover, beginning on page 55, an example is taken by applying the microprogram flow chart of Figure 14a and actually microprogramming the ROM in the control store of Figure 13b to selectively generate the signals via the means claimed in the second claim for controlling certain predetermined tasks and functions. For a further discussion on this topic, see pages 10 and 11 of the Brief of Appellant. See also pages 12 and 13 of the Brief of the Appellant which gives further information as to the nature of instructions and how they are embodied in computers.

The Examiner has further mischaracterized the nature of an operating system endowing it with attributes it does not possess such as "registers." On page 5 of the Examiner's answer, lines 2-6, he states "Applicants contend that their 'firmware' invention when loaded into the registers of the operating system result in an invention having structural aspects and hardware limitations." Sippl and Sippl in their *Computer Dictionary and Handbook*, define "operating system" on page 300, as ". . . a part of a software package (program or routine) defined to simplify housekeeping as input/output procedures, sort-merge generators, data-conversion routines, or tests." Clearly an "operating system" does not have (nor can it have) registers. The correct statement is that when firmware is *incorporated* into the computer system (not operating system) it results in an improved hardware structure (Figure 13b) which cooperates and functions with other hardware structures (Figures 12, 13a) to provide new and unobvious capability to the computer system. The operating system enters into the picture only peripherally in that it can now communicate via this hardware invention with processes executing on the computer system, without programmer intervention.

Having established that the subject matter of an invention claimed must be treated as a whole, the principle enunciated by the Supreme Court in Gottschalk v. Benson 1515 USPQ 673 (1972) as interpreted by CCPA decisions is inapplicable to this case. The latest case

(of which Appellants are aware) decided on July 28, 1977 by the USCCPA, is "In the Matter of the Application of Gerald Waldbaum", re Patent Appeal 76-690. Although the Board of Appeals was affirmed since the claims were method claims in this case, the interpretation of Benson and Tabbot enunciated therein, is nonetheless significant. In the Court's opinion the preemption or nutshell rationale of Benson and Tabbot is applied to affirm the Board of Appeals. The Court defines the "nutshell rationale" as follows" . . . a patent on the claims in issue "would wholly preempt the mathematical formula and in practical effect would be a patent on the algorithm itself' (Benson supra at 72, 175 USPQ 676)." The Court further sheds light on In Re Christensen 178 USPQ 35 (CCPA 1973), wherein the Court states ". . . although the claims were limited to a specific application of technological significance, they were, nonetheless, directed to a *calculation* and would have preempted use of the algorithm in making the calculation."

The instant invention does not involve nor does it claim a method, process, or algorithm directed to a calculation. It is hardware in a computer, which permits direct communication of an operating system with processes executing on the computer system. It is disclosed as such. It is claimed as such. All that could be preempted when all the elements of a claim, with all their limitations, are taken as a whole, is the *hardware* defined by the claims that permits such communication. It does not preempt other hardware, not so defined by the claims that communicate between an operating system and process executing on the computer system. It does not preempt any algorithm for such communication because none is disclosed and none is claimed. It does not preempt any algorithm directed to a calculation. As a matter of fact, the claims when properly read considering the elements as a whole and how they function and cooperate to achieve a solution to a problem plaguing the prior art, particularly point out and distinctly claim subject matter which does not fall under the purview of 35 USC 100 or 101.

It is well established that an invention is defined by the claims interpreted in light of equivalents and the specification. It is further well established that a claim can define an invention in terms of a combination of old elements which are so related one to the other, and function one with the other to produce a new and unobvious result. It is also established that such a claim, or any claim for that matter, must be interpreted as a whole, including many times the solution to the problem which eluded the prior art. To dissect such a claim and eliminate elements from consideration on the pretext of being old in the art, and in order to force fit it in a category completely alien to its own attributes, is not supported by either statutory or case law.

It is submitted that every invention is *based* on a fundamental law, basic principle, mathematical formula, or algorithm. 35 USC 101 proscribes patenting the fundamental law, basic principle or mathematical algorithm on which the invention is based. To do otherwise would preclude meritorious future inventions based on such fundamental law, basic principle or mathematical algorithm. Benson & Tabbot admonishes us that when the language of a claim is so broad so as to preempt all uses of an algorithm, it is in effect a claim to the algorithm itself. By arbitrarily eliminating essential and limiting elements from consideration even though they be old in the art, a claim can arbitrarily be interpreted so broadly so as to read on the algorithm itself. This was not done in Benson & Tabbot and it should not be permitted here.

CONCLUSION

The sole issue in this appeal is whether or not the subject matter claimed is statutory under 35 USC 101. Clearly the subject matter of the invention is in the technological arts. It is not a method, technique or process nor claimed as a method, technique or process. It is not an algorithm nor is an algorithm claimed. It does not preempt any or all algorithms utilized to communicate between an operating system of a computer system and processes executing on the computer system. It is not an

algorithm directed to a calculation. The subject matter of the invention as disclosed and claimed is hardware. Clearly the subject matter of the invention is patentable under 35 USC 101, and it is respectfully requested that the Examiner be reversed, the claims allowed, and the application passed to issue.

ORAL HEARING REQUEST

Appellants have requested an Oral Hearing, accordingly two extra copies of this Reply Brief are provided in accordance with Rule 192.

Respectfully submitted,

/s/ Nicholas Prasinos
NICHOLAS PRASINOS
Registration No. 24,791

OPINION AND DECISION OF BOARD OF APPEALS, SEPTEMBER 20, 1978.

Before Burns and Spencer, Examiners-in-Chief, and Craig,

Acting Examiner-in-Chief.

Burns, Examiner-in-Chief.

This is an appeal from the final rejection of claims 1-6, constituting all of the claims presently in the case.

The subject matter concerns computer "firmware." [1] In their brief appellants characterize their invention in the following words:

> ". . . More specifically, the invention concerns a combination of hardware including a model-independent instruction means which is implemented in firmware, for adding capability and increased functionality to the computer system. As defined by the Applicants' claims, the present invention is directed to a combination of hardware structures controlled by a microprogrammed control system which provides for the generation of signals which cause the creation of an image of a communication center at another location, and for the generation of signals which causes the interchange of the information between the communication center and its image."

More particularly, the problem addressed and appellants' solution thereto may be discerned from the following selected portions of appellants' disclosure. From page 4, beginning at line 15:

> "High performance computing systems find it necessary to store portions of the system base in internal scratchpad registers instead of main memory,

---

[1] The term has been defined as follows:

firmware-Logic circuits in read-only memory that may be altered by the software under certain circumstances, (*Computer Dictionary and Handbook*, Sippl and Sippl, published by Howard W. Sams & Co., Inc., The Bobbs-Merrill Co. Inc., 2nd Ed., 1972.)

and hence inaccessible and invisible to software running in the system. Accordingly, a problem results when software (i.e. the programmer) wants to change positions or change information of the system base which happens to be located in these internal central processor scrathpad [sic] registers which software cannot access."

From the same page, beginning at line 24:

"The problem has been solved in one of two ways. The first method is to simply re-initialize the entire system, loading the system base with different contents as a result of a new initialization process. This technique is extremely time-consuming and quite undesirable.

"A second technique for solving the problem is to have the software employ special model-independent properties of the particular machine on which it was running, to change the system base. This results in the very undesirable property of completely model-dependent software—a feature which certainly does not serve to endear a computer manufacturer to a customer.

"What is needed therefore is a technique of changing information in the system base both in main memory and internal scratchpad registers making the system base in scratchpad memory accessible to software but not visible to software i.e. cannot be read by the software."

From page 5, beginning at line 26:

"A model-independent instruction in firmware is provided to change, upon its execution, information in the system base which is located in both main memory and internal scratchpad registers.

"The system base located at a location specified by the boundary address register BAR in main memory, is copied into a new location in main memory with predetermined portions of its information changed. This is readily accomplished by the operating system. . .

. . .

> The switch system base instruction then transfers the information which is in the system base at the new location into the system base at the location specified by BAR and also that portion of the system base located in scratchpad memory."

We reproduce claim 1 as representative, however, in so doing, we interleave indented explanation (identified parenthetically with capital letters A to D) of the respective portions of the claimed subject matter, the explanation being derived from pages 9 and 10 of appellants' brief

1. In a multiprogramming computer system having a main memory, a central processing unit (CPU) coupled to said main memory, said (CPU) controlling the state of a plurality of groups of processes being in a running, ready, wait or suspended state, said computer system also having scratchpad registers being accessible to an operating system for controlling said multiprogramming computer system, a data structure for storing coded signals for communicating between said processes and said operating system, and said scratchpad registers, said data structure comprising:

(a) first means in said data structure and communicating with said operating system for storing signals indicative of an address for a selected one of said processes;

> (A) ". . . data structure claimed is comprised of the first means which is the structure of the running process word register at BAR+56 of the system base structure of Figure 6. The system base structure is described with the particularity of 35 USC 112 beginning on page 35 through 41. Its use is further described beginning on page 41. The running process word register structure is further shown with particularity on Figure 11j and is described with particularity beginning on page 39 of the specification . . ."

(b) second means in said first means for storing coded signals indicating priority of said selected one of said processes in relation to others of said processes for obtaining control of said CPU when ready;

(B) ". . . The second means which is an element within the first part of the running process word register and is identified with reference to numeral 1129. . ."

(c) third means in said data structure and communicating with said operating system, for storing coded signals indicative of an address for a selected one of said plurality of groups of processes; and,

(C) ". . . The third means within the data structure is the J table word means and the G table word means at BAR and BAR+4 on Figures 11a and 11b and described in the specification. . ."

(d) forth means coupled to said data structure and said scratchpad registers, for generating signals causing the changing of information in said data structure and said scratchpad registers.

(D) ". . . Finally, the fourth means claimed is shown on Figures 13a and 13b is a central store as microprogrammed by the microprogram as disclosed by the flow charts of Figures 15b and 15c. Details of the control store which is microprogrammed are shown on Figure 13b (it should be understood that control stores are old in the art and persons of ordinary skill in the art can microprogram them when provided with the fiow charts of the microprogram and this results in the firmware. . ."

THE REJECTION

All of the claims have been rejected under 35 USC 101 as drawn to non-statutory subject matter. The examiner cites *Gottschalk* v. *Benson*, 409 US 63, 175 USPQ 673 (1972).

In his answer, beginning at page 5, line 6, the examiner explains his position in part as follows:

". . . Claim 1 has been couched in apparatus format and recites first means through fourth means. The first means through third means set forth means for storing signals and read upon registers or memory

locations within the operating system, already shown to be old in the art. The function of or attribute of the signals stored in the first, second, and third means for storing may be classified as indicating an address for a selected one of a plurality of processes, indicating a priority of said selected process relative to other processes, and indicating an address of a selected one of the plurality of processes, respectively. There is clearly no novelty to be found here for distinguishing over that which is deemed to be old in the art.

"The fourth means is stated as means for generating signals causing the changing of information in the data structure and scratchpad registers and is categorized by applicants' brief, page 10, as a control store as microprogrammed by the microprogram of Figures 15b and 15c. The use of control stores in computer systems has been shown to be old in the art in the listing of publications and U. S. Patents in paragraph 19 of Appendix I, amendment filed August 11, 1976. In fact, paragraph 22 of the Appendix I correctly indicates that microprogrammed computer processors including microprogrammed control units are old in the art and suggests that the systems shown in the two cited patents are capable of practicing or implementing the instant invention. . . .

. . . .

Applicants' invention resides in an instruction in firmware which changes, upon execution, information in the system base. The claim is thus reciting prior art coupled with subject matter which the U.S. Supreme Court has found to be non-statutory in *Benson*. . . ."

The appellants' argument is to the effect that only hardware, that is, structure, has been recited wherefore the *Benson* decision is inapposite. They characterize the examiner's rationale as procrustean "in forcing an arbitrary conclusion or solution to the facts," brief, bottom of page 8.

OPINION

In order to determine what appellants regard as their invention, it is necessary to consider the claimed subject matter as a whole, *In re* Noll, 545 F.2d 141, 191 USPQ 721 (CCPA, 1976), *In re* Chatfield, 545 F.2d 152, 191 USPQ 721 (CCPA, 1976). With this caveat in mind it is nevertheless appropriate to consider the features, both novel and otherwise, within the claims, as did the Court in *Parker* v. *Flook*, 437 US ——, 198 USPQ 193 (1978), an approach "not at all inconsistent with the view that a patent claim must be considered as a whole."

Analysis of Claimed Subject Matter

First, we observe that neither the term "firmware" nor an apparent equivalent appears in any of the claims, cf. 35 USC 112, last sentence. The various recited "means" for storing, communicating and generating are not to be construed as limited to firmware.

In an amendment filed August 11, 1976 (Paper No. 4), appellants amended their specification with APPENDIX I providing numbered items which refer to prior art features, to which APPENDIX we will have occasion to refer.

Our comments on the elements recited in the claims will follow their order of recitation within each claim.

Claim 1

For the main memory, appellants in APPENDIX I, (11) (a) appear to rely for disclosure upon references 2-7 found on page 2 of the specifications. References 2-6 upon inspection appear to show a mix of block diagrams directed to both hardware and software. Reference 7 shows only flow charts without a suggestion of hardware or apparatus structure. To the extent that references 2-6 disclose structure or hardware it would appear that, as such, these references demonstrate that memory *per se* is old. It does not appear that appellants have argued otherwise, rather, appellants urge that they claim ". . . new, novel and unobvious *arrangement* of these struc-

tures . . .," brief, page 9, beginning at line 4, (our emphasis).

As for the central processing unit (CPU), no specific reference has been identified in APPENDIX I by this term, nevertheless, item (22) thereof describes microprogrammed computer processors employing microprogrammed control units "with which the instant invention may be practiced."

Concerning the "data structure" of the claim, page 11, beginning at line 18, of the specification characterizes this feature of being composed of data formats, software visible registers, and instruction formats. The first and last of these three features are described in paragraphs A of specification page 11, line 21 and J of page 14, line 15. Nothing in this description of formats suggests hardware. It would seem that reliance must be placed on the software visible registers, the second feature, as described in paragraph E, page 12, beginning at line 24 of the specification. Appellants appear to rely at prior art for such features upon patents described in (12) and (13) in APPENDIX I.

Appellants have characterized the "first means" as the running process word register within the data structure. Save for the actual word stored which, it would appear, will change depending upon the nature of the data processing undertaken, this feature consists of space in memory (BAR+56 as shown in Figure 6) as part of the system base.

From the foregoing it would seem that the data structure consists of at least one software visible register, and the first means in said data structure constitutes that register (or a portion of it) with a particular word stored therein, an address. It does not appear that the "first means" constitutes novel hardware distinct from the registers designated above representing the data structure for which appellants have referenced prior art.

Similarly, for the second and third means of the claim, so much thereof as may be characterized as hardware would seem to have antecedents in the prior art registers as part of memory described above for the data structure.

The fourth means of the claim has been designated by the appellants as a central store, known to be old, brief, page 10, line 8. It is there characterized:

> ". . . as microprogrammed by the microprogram as disclosed by the flow charts of figures 15b and 15c . . ."

Appellants have buttressed this by the prior art references in APPENDIX I (19), (20) and (21).

From the above we conclude that the components recited in claim 1 appear to be old in the art constituted as registers as part of memory (the latter being hardware) save for the microprogramming provided by element (d) of the claim, the fourth means. Any "specific new, novel and unobvious arrangement of these structures," brief, page 9, beginning at line 4, would appear to be provided by the microprogramming of Figures 15b and 15c, which together with its attendant memory hardware appears to constitute firmware.

Claims 2-6

As to these claims, analysis of the nature set forth above in connection with claim 1 leads to a similar conclusion. We would only add that, with respect to claim 4, appellants have made reference to arithmetic and logic units in Appendix I(2) (a). The element characterized as "hardware gate" found in claim 5 appears also to be part of memory, for which appellants have provided prior art references, under the control of the programming shown in Figures 15b and 15c, see specification page 64, beginning at line 26 et seq., of the specification.

Authorities

Appellants in their brief rely upon *In re* Noll, *supra*, and *In re* Chatfield, 545 F.2d 152, 191 USPQ 730 (CCPA, 1976). In their reply brief, appellants comment upon *In re* Waldbaum, 559 F.2d 611, 194 USPQ 465 (CCPA, 1977) and urge the significance of the court's language in distinguishing the Waldbaum subject matter from that in *In re* Christensen, 478 F.2d 1392, 178 USPQ 35 (1973). Appellants there maintain that "the instant in-

vention does not involve nor does it claim a method, process or algorithm directed to a calculation." We disagree. It appears to us that Waldbaum, if anything, supports the examiner's position, in that the Court proscribed claims "which are directed to methods for controlling the operation of a data processor," basing its decision upon the *Benson* decision. It would seem that appellants' claims are similarly directed.

We recognize that method claims were under consideration in Waldbaum whereas here the claims are in apparatus format. In *In re* Freeman, 197 USPQ 464 (CCPA, 1978) the Court found such a circumstance to be of no moment:

> "Though a claim expressed in 'means for' (functional) terms is said to be an apparatus claim, the subject matter as a whole of that claim may be indistinguishable from that of a method claim drawn to the steps performed by the 'means.'
>
> "We agree with the solicitor's contention that if allowance of a method claim is proscribed by Benson, it would be anomalous to grant a claim to apparatus encompassing any and every 'means for' practicing that very method."

We are aware that the Waldbaum claims included characteristic language obviously related to calculating and mathematical problem solving, thus (from *Waldbaum* claim 1):

> ". . . a method . . . to *count* the number of busy lines. . . . . *comparing* means . . . to derive the *number of 1's* in said data word." (Emphasis added.)

That appellants' claim are absent similar characteristic language does not make the functions attendant the "means" of appellants' claims any less mathematical or less related to an algorithm within the meaning assigned that term by the USSC in *Benson.*

The specification states, page 12, line 2:

> "All data are in binary form, but may be interpreted as binary, decimal or alphanumeric."

Since digital computers normally operate in some numerical radix, binary, binary coded decimal, or the like, we consider the operation of appellants' claimed invention to be mathematical.

Every operation performed in appellants' invention as claimed involves the accommodation of data and instructions to the size of the registers in memory, and to the positional assignment to the registers in memory by the use of some numerical measure or quantity effected by way of electrical signals. In whatever form the instructions employed in appellants' invention are characterized, numerical or otherwise, we think it is accurate to say that the operation of appellants' structure is mathematical and that the instructions constitute a procedure which is algorithmic in character, to the same degree as that of the Waldbaum structure and that satisfactory operation of the apparatus claimed represents the successful solution of a mathematical problem falling within the definition of algorithm supplied in *Benson* and reiterated in *Flook*.

Here the method of calculation is to be used in the internal operation of the computer system and the claims recites such an end use by means of the functions recited for the various means. A claim for an improved method of calculation, even when tied to a specific end use, is unpatentable subject matter under 35 USC 101, *Flook*, *supra*.

SUMMARY

We agree with the examiner. The claims under consideration, when each is considered as a whole, recite apparatus as a matter of form only, and couple such apparatus, which appellants have indicated as old in the art, with subject matter, namely, programming, which is nonstatutory under the *Benson*, *Christensen*, *Waldbaum* and *Flook* cases, *supra*. Employing appellants' metaphor, the claims at bar, we think, extend beyond the procrustean statutory bed.

The decision of the examiner is affirmed.

AFFIRMED

/s/ [Illegible]
Examiner-in-Chief

/s/ [Illegible]
Examiner-in-Chief

/s/ [Illegible]
Examiner-in-Chief
(Acting)

BOARD OF APPEALS

NOTICE AND REASONS OF APPEAL TO UNITED STATES COURT OF CUSTOMS AND PATENT APPEALS, NOVEMBER 17, 1978.

November 13, 1978

Commissioner of Patents & Trademarks
Washington
District of Columbia 20231

Dear Sir:

Applicants in the above-identified appeal hereby serves notice under 35 USC 141 and 142 of their appeal to the Court of Customs and Patent Appeals from the decision of the Board of Appeals in Appeal No. 329-86 rendered on September 20, 1978. The following are assigned as reasons of appeal:

The board erred in holding that:

1. It is appropriate to consider the features, both novel and otherwise within the claims, when only a non-statutory subject matter rejection under 35 USC 101 was at issue.
2. Any specific new, novel and unobvious arrangement of the combination claimed would appear to be provided by the microprogramming of figure 15b and 15c, which together with its attendant memory hardware appears to constitute firmwave.
3. Since digital computers normally operate in some numerical radix, binary, binary coded decimal, or the like, we consider the operation of appellants' claimed invention to be mathematical.
4. In whatever form the instructions employed in appellants' invention are characterized, numerical or otherwise, we think it accurate to say that the operation of appellants' structure is mathematical and that the instructions constitute a procedure which is algorithmic in character, to the same degree as that of Waldbaum structure and that satisfactory operation of the apparatus claimed represents the successful solution of a mathematical

problem falling within the definition of algorithm supplied in Benson and reiterated in Flook.

5. Here the method of calculation is to be used in the internal operation of the computer system and the claims recite such an end use by means of the functions recited for the various means.
6. A claim for an improved method of calculation, even when tied to a specific end use, is unpatentable subject matter under 35 USC 101.
7. The claims under consideration, when each is considered as a whole, recite apparatus as a matter of form only, and couple such apparatus, with subject matter, namely, programming, which is non-statutory under the Benson, Christensen, Waldbaum and Flook cases.

The board erred in not reversing the examiner's holdings that:

1. Claims 1-6 are deemed properly rejected under 35 USC 101 as drawn to non-statutory subject matter.
2. Applicants' invention resides in an instruction in firmware which changes, upon execution, information in the system base. The claim is thus reciting prior art coupled with subject matter which the U.S. Supreme Court has found to be non-statutory in Benson.
3. The claims are reciting prior art coupled with subject matter which the U.S. Supreme Court has found to be non-statutory in Benson.
4. Applicants' contribution to the art resides in the specific instruction found within the memory locations and not the *combination* of three memory locations labeled first, second, and third means.

Respectfully submitted,

/s/ Nicholas Prasinos
Nicholas Prasinos
Attorney for Applicants
Registration No. 24,791

REJECTED CLAIMS.

1. In a multiprogramming computer system having a main memory, a central processing unit (CPU) coupled to said main memory, said (CPU) controlling the state of a plurality of groups of processes being in a running, ready, wait or suspended state, said computer system also having scratchpad registers being accessible to an operating system for controlling said multiprogramming computer system, a data structure for storing coded signals for communicating between said processes and said operating system, and said scratchpad registers, said data structure comprising:

(a) first means in said data structure and communicating with said operating system for storing coded signals indicative of an address for a selected one of said processes;

(b) second means in said first means for storing coded signals indicating priority of said selected one of said processes in relation to others of said processes for obtaining control of said CPU when ready;

(c) third means in said data structure and communicating with said operating system, for storing coded signals indicative of an address for a selected one of said plurality of groups of processes, and,

(d) fourth means coupled to said data structure and said scratchpad registers, for generating signals causing the changing of information in said data structure and said scratchpad registers.

2. In combination with a communication center located in a predetermined area of a computer main memory the computer having a central processing unit (CPU) for executing plurality of processes that compete with each other for control of said CPU, operating system means coupled for providing information to said communication center, and a scratch pad memory in said CPU which stores, under control of said operating system means, portions of information located in said communication center, an improved system controlled by one of said processes for changing information in said communica-

tion area and in said scratchpad memory, said improvement comprising:

(a) first means for generating signals causing the creation of an image of said communication center at another location in said main memory;

(b) second means responsive to said operating system means for generating signals causing a change of the information in said image, resulting in a new image; and,

(c) third means, coupled for being responsive to said first and second means and to one of said processes for generating signals for causing a transfer of the new image to said communication center and to said scratchpad memory.

3. The system as recited in Claim 2 wherein said means is a switch system base instruction in firmware.

4. The system as recited in Claim 3 further including an arithmetic and logic unit having an auxiliary memory said arithmetic and logic unit coupled for communicating with said first, second and third means.

5. The system as recited in Claim 4 wherein said auxiliary memory includes a hardware gate which has a locked and unlocked state said unlocked state permitting internal interrupts or external exceptions, and said locked state not permitting internal interrupts or extral exceptions.

6. The system as recited in Claim 5 wherein said hardware gate is locked by said switch system base instruction when executing and unlocked by said switch system base instruction when it terminates execution.

SUPREME COURT OF THE UNITED STATES

No. 79-855

SIDNEY A. DIAMOND, COMMISSIONER OF PATENTS AND TRADEMARKS, PETITIONER

v.

JOHN J. BRADLEY and BENJAMIN S. FRANKLIN

ORDER ALLOWING CERTIORARI

Filed March 17, 1980

The petition herein for a writ of certiorari to the United States Court of Customs and Patent Appeals is granted. The case is set for oral argument in tandem with No. 79-1112, *Diamond* v. *Diehr, et al.*

☆ U. S. GOVERNMENT PRINTING OFFICE: 1980 317794 282